CONTRACTOR REPORT ARPAD-CR-83002

DEVELOPMENT AND QUALIFICATION OF COMPUTER-AIDED ASSESSMENTS (CAS)

R. LARRY WILLIAMS
KAMAN SCIENCES CORPORATION
1500 GARDEN OF THE GODS ROAD
COLORADO SPRINGS, COLORADO 80933

NICHOLAS ZUCK
PROJECT LEADER
ARRADCOM

FEBRUARY 1983

US ARMY ARMAMENT RESEARCH AND DEVELOPMENT COMMAND
PRODUCT ASSURANCE DIRECTORATE
DOVER, NEW JERSEY

The views, opinions, and/or findings contained in this report are those of the author(s) and should not be construed as an official Department of the Army position, policy, or decision, unless so designated by other documentation.

The citation in this report of the names of commercial firms or commercially available products or services does not constitute official endorsement by or approval of the U.S. Government.

Destroy this report when no longer needed. Do not return to the originator.

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: Contractor Report ARPAD-CR-83002	2. GOVT ACCESSION NO.
3. RECIPIENT'S CATALOG NUMBER	
4. TITLE (and Subtitle): DEVELOPMENT AND QUALIFICATION OF COMPUTER AIDED ASSESSMENTS (CAS)	5. TYPE OF REPORT & PERIOD COVERED: Final Report
6. PERFORMING ORG. REPORT NUMBER: DRDAR-QAM-72-82	
7. AUTHOR(s): R. Larry Williams, Kaman Sciences Corporation Nicholas Zuck, Project Leader, ARRADCOM	8. CONTRACT OR GRANT NUMBER(s): DAAK10-81-C-0316
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Kaman Sciences Corporation 1500 Garden of the Gods Road Colorado Springs, Colorado 80933	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS: ARRADCOM, TSD STINFO Div (DRDAR-TSS) Dover, NJ 07801	12. REPORT DATE: February 1983 13. NUMBER OF PAGES: 145
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office): ARRADCOM, PAD Nuclear Systems Div (DRDAR-QAN-R) Dover, NJ 07801	15. SECURITY CLASS. (of this report): Unclassified 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Computer analysis
RAM assessments
Reliability assessments
Safety assessments
Maintainability assessments
Fault trees
Confidence bounds
Probabilities

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

Maximum utilization of computer-aided assessments (CAS) techniques and proper understanding of nuclear safety and reliability assessments can be achieved by the use of a program called "GO".

This document is a basic, yet substantive, qualification of the GO methodology and its computer software. Principal emphasis is placed on the event-tree concept upon which GO is based and on the use of the defined operator types (cont)

20. ABSTRACT (cont)

(which represent components) to develop system models. The use of GO software to manipulate models and component data to generate the probabilities of occurrence of system events (reliability, safety, availability) is explained and exemplified. Seven sample systems are analyzed.

A qualification of the fault finder and effect evaluation software is provided. The fault finder capability identifies component failures which singly or in combination cause system failures. The effect evaluation capability develops confidence bounds on system estimates as a function of data uncertainties.

ACKNOWLEDGEMENT

The present GO computer programs, KSC GO, Version 1.0, are the distillation of work and experience over the last fifteen years. Wilson Y. Gateley and R. Larry Williams of Kaman Sciences Corporation (Colorado Springs, Colorado), hereafter referred to as KSC, have continued to form the nucleus of GO-related research efforts. In addition there have been many contributions from other members of the KSC staff as well as from individuals from other organizations. Present and former members of the KSC staff who have materially aided the GO development are Dan W. Stoddard, Noel J. Becar, Gale B. Curtis, Donald C. Wood, and Joyce A. Wenger. Other individuals who have aided directly or indirectly include Nick Zuck of ARRADCOM, Don Emon of DOE, Boyer B. Chu of EPRI, and Roberta Galante of TVA. Participants in the many GO training seminars have helped structure the presentation and provided suggestions for new features. We gratefully acknolwedge the help of these contributions in developing the GO methodology.

SUMMARY

Applications of the GO Methodology

The GO methodology has been available since 1967. It was originally developed under U.S. Army contracts to analyze the safety and reliability of nuclear weapons and missile systems. Using the GO methodology comprehensive safety and reliability analyses of the SPRINT, SPARTAN, NIKE HERCULES, HONEST JOHN, LANCE, PERSHING 1a, MADM, M454, M422, and M753 Army nuclear weapon systems have been performed. Similar studies have been performed on the POLARIS, POSEIDON and TRIDENT weapon systems for the Navy.

In the late 70's, GO capabilities were expanded under EPRI and utility sponsorship to analyze the safety and availability of conventional and nuclear power plants. The Kaman Sciences Corporation (KSC) team which developed the GO methodology has applied it to analyze the Three-Mile Island Unit 2 SCRAM System, the Fort St. Vrain SCRAM System, the Dresden 2 Emergency Diesel Generator System, the Utah Power and Light Huntington 2 plant, the Sequoyah and Bellefonte Auxiliary Feedwater Systems, etc.

KSC also demonstrated the GO capability to model and analyze an entire nuclear plant - the TVA Sequoyah 1.1 MWe - integrating the combined effects of approximately 60 systems and 10,000 components to assess plant availability and probabilistic risks. The direct accommodation of system interactions in one integrated model was one significant feature of this study. This capability permits the identification of common mode faults from electrical, cooling water and instrument air systems which may affect many systems simultaneously.

Application of the GO methodology involves five steps: (1) gaining a thorough understanding of the operation of the system to be modeled; (2) defining system success and failure states; (3) representing physical components and logical operations with GO symbols; (4) interrelating the GO symbols representing physical components and logical operations into a GO model representing the engineering functions of a component/subsystem/system; and (5) executing the GO computer codes to quantify system performance using the system GO model. The methodology is capable of (1) evaluating system reliability/availability, (2) identifying system fault combinations, (3) constructing confidence bounds on the numerical results, and (4) ranking the impact of constituent elements on system performance.

Significant Features of GO

The GO methodology possesses some significant features which differentiate it from other probabilistic system assessment techniques. Some of these modeling characteristics are summarized below:

(1) GO models, which are composed of a collection of operator symbols, are usually developed by following normal process flow. The models are constructed from engineering blueprints and flow diagrams by using representative GO symbols for physical equipment (valves, motors, pumps, etc.). The inputs to, and the outputs from, these elements are then linked together to form a GO model. The similarity between a GO model and its corresponding engineering drawing makes it easy to validate and interpret the model.

(2) GO methodology models system functions and operational logic. Thus, a GO model contains all possible system operational states of the constituent elements. The effects of system design enhancement, procedural alterations, etc., on the various operational states can be evaluated.

(3) Alterations and updates to a GO model are readily accomplished. Adding or deleting components or altering the logical combinations of equipment can be accommodated without extensive alterations to the basic model structure of the computer input information. Changes to system boundary conditions can be easily accommodated by changing the data input rather than by changing the model itself. The block modeling feature using supertypes also enhances the ease with which model alterations can be effected.

(4) The identification of fault sets is a powerful technique to identify various equipment failure combinations which preclude successful operation. In GO terminology the term "fault set" is used as being more general than cut set because GO models often incorporate more than just two operational modes, e.g., success, failure, and premature. Having developed a GO model, the fault sets for any operational state of a system can be automatically generated.

(5) The GO computer codes utilize a system GO model and the definitions of GO symbols to quantify system performance. The calculating algorithms in the computer codes employ a truncation procedure whereby events of least significance, as measured by their probabilities of occurrence, are discarded. The truncation procedure enhances the computational efficiency of the methodology. The error introduced by truncation is known and can be controlled.

Additional characteristics about the GO methodology which experience has shown to be significant are that it:

(1) calculates results to any desired accuracy and precision,

(2) can include in its models and measure the significance of all system elements and characteristics,

(3) relates component characteristics to system characteristics using classical theory,

(4) provides diagnostic capabilities to analyze the cause of any peculiar, unusual, or significant system event,

(5) models and rigorously documents system configuration providing traceability and repeatability,

(6) is easy to use by engineers not specialized in statistics and reliability disciplines,

(7) is an accepted, standardized procedure having widespread application,

(8) is not limited to nor restricted by a simplified "black box" block-diagram approach which ignores many dependencies,

(9) replaces error-prone procedures such as equation writing and other manual techniques,

(10) suppresses unnecessary intermediate analysis details which are difficult to interpret (e.g., lengthy equations, voluminous fault trees),

(11) standardizes baseline models providing the basis for comparisons, tradeoffs, value studies and uniformity of interpretation,

(12) requires less analyst time, and less skilled analysts, than other available procedures and is, consequently, more efficient and cost-effective than similar procedures.

Validation of GO Methodology

The GO methodology has been repeatedly compared with fault tree methodology and has demonstrated that it provides comparable, and more comprehensive, results. Some of the studies involving such comparisons are:

(1) Comparison of Reliability Evaluation Methodologies, K74-83U(R), 3 July 1974, Gale B. Curtis (Nick Zuck - ARRADCOM)

(2) A Comparison of Results from the GO Methodology and Fault Tree Analysis, NUREG-K77-38U(R), 31 August 1977, D. E. Wood and Noel J. Becar (James W. Pitman - NRC)

(3) Demonstration of a Reliability Assessment Methodology as Applied to Nuclear Power Plant Systems, K74-30U(R), 9 April 1974, R. L. Williams and Noel J. Becar (Donald Eamon - DOE)

(4) GO Evaluation of a PWR Spray System, EPRI 350-1, August 1975, W. T. Long, R. L. Williams (Alexander - EPRI)

(5) Audit and Verfication of Existing RAM Assessments, (1979) (U.S. ARRADCOM Contract - N. Zuck)

Acceptance of GO Methodology

The GO methodology is becoming a standard assessment procedure for analyzing the safety and reliability of military and space weapon and communication systems. It is similarly becoming a standard availability and probabilistic risk assessment procedure for nuclear and conventional power plants.

The Army recently completed a series of eight GO training seminars in which 138 Army and contractor personnel were provided hands-on experience applying the GO software to perform assessments. Similar training has been conducted for utilities beginning with a training session in Dallas, Texas, in July 1980.

A number of nuclear plant probabilistic risk assessments are presently being conducted using the GO methodology, e.g., Sequoyah, Bellefonte. Availability assessments using GO have also been performed on Midland, Sequoyah, etc.

The recent (April 1982) draft Procedures Guide for Performing Probabilistic Risk Assessments of Nuclear Power Plants, published by the Nuclear Regulatory Commission, contains several pages addressing the strengths of the GO procedure. The Electric Power Research Institute is funding GO-related applications at the rate of approximately a million dollars a year.

KSC has received inquiries from scientists in more than twenty foreign countries for more information about the GO methodology.

Because of its simplicity, ease of use, generality, comprehensive capabilities, efficiency, and lower cost of application, the GO methodology is becoming a standard state-of-the-art method for performing system assessments.

Use of the CAS/GO methodology by trained analysis permits refined and comprehensive system assessments accounting for all dependencies that were intractable several years ago. Its use also highlights unfounded assertions of prior studies and focuses on the specific data required to generate valid performance measures. Assessment engineers and responsible managers should be knowledgeable of this latest development in a computerized analysis methodology.

Because the procedure is so powerful, managers and engineers involved in planning for, designing, fabricating, fielding, and assessing the performance of any major system, should be sufficiently knowledgeable of the GO methodology to take advantage of its unique capabilities to aid the decision-making process. GO training for responsible personnel should be provided and the GO software and its documentation should be acquired.

FOREWORD

Background

An audit and verification performed on existing RAM assessments indicated numerous differences in applying various mathematical assessment techniques and developing special system models. Of primary importance, the audits demonstrated the need for a baseline computer-aided assessment methodology to standardize RAM assessments and to permit the development of more accurate system models.

In this respect, the CAS procedures using KSC "GO" methodology were employed to perform RAM audit assessments and, as a result, uniquely demonstrated a qualified computer-aided assessment technique which is highly cost effective and includes many advantages.

The differences between the CAS GO methodology and the failure or success equation block diagram and fault tree approaches used previously in RAM assessments are significant. Some of the more significant capabilities provided by the computerized GO methodology are:

1. Calculates results with accuracy and precision
2. Includes and measures the significance of all elements and characteristics
3. Relates component characteristics to system characteristics using classical theory
4. Provides diagnostic capabilities
5. Permits traceability and repeatability
6. Is not limited to nor restricted by the "black box", simplified block-diagram approach
7. Replaces error-prone procedures such as equation writing and other manual techniques
8. Suppresses unnecessary intermediate analysis details which are difficult to interpret such as lengthy equations
9. Provides standardized or baseline models as a basis for comparisons, tradeoffs, value studies, and uniformity of interpretation.

Development

The GO software was initially developed by KSC scientists Bill Gateley, Larry Williams, and Dan Stoddard under Army funding in 1967. With the passage of time the codes have undergone almost continual development and refinement by the KSC originators. In the early 70's, a number of new features were

added - data consistency checks, new operators 11-15, supertypes, sensitivity studies, etc. In the fall of 1975, KSC conceived the fault-finder and confidence interval algorithms. These were subsequently implemented with internal IR&D funds, continuing Army, Navy, and NRC funding and EPRI funding.

Because of the unique formulation of the GO methodology and its modeling approach, its use and acceptance are widespread. Users of various versions of the codes include the Army Research and Development Command, Eglin Air Force Base, Boeing-Wichita, University of Washington, Bechtel Power Company, Brown & Root, Inc., Utah Power and Light Company, Tennessee Valley Authority, Cincinnati Gas and Electric Company, San Diego Gas and Electric Company, Houston Power and Light Company, Pickard, Lowe and Garrick, Inc., etc. The latest version, KSC GO Version 1.0, is also on the Control Data Corporation (CDC) CYBERNET timeshare system.

With many users and increased application there is a continuing need for code updates and modifications. There is also need for continuing support and standardized versions of the codes which KSC as the originators and most experienced users continues to provide.

Since the original creation of the GO codes in 1967 there have been 24 documented KSC versions of the codes as shown below.

VERSIONS OF THE GO CODES

DATE	NAME	CHARACTERIZATION
APR 68	GOMAR68	Eleven Logical Operators, Hash Addressing
JUN 68	GOJUN68	---
APR 69	GOAP69	Type 9 & 11 Kind Data Changed, Sensitivity Runs, Format-free Data, Modular Programs, Time Points up to 9999 permitted
MAY 69	GOMAY69	Use of two computer words to store more active signals and handle larger problems
AUG 70	GOAUG70	---
1971	RANGO	Randomized GO, Component Beta Distributions
1972	GOCHK72	---
APR 74	GOAPR74	Data Checks, Signal Table
1974	XGO	100 Active Components, Automatic Signal Deletion, Extensive Error Checking, Perfect Component Case, Automatic Array Size Optimization, PMIN Reset, Types 5 & 11 Combined
1975	Version B	---

VERSIONS OF THE GO CODES (Continued)

DATE	NAME	CHARACTERIZATION
16 FEB 76	Version C	New Operators 11-15, Multiple Type 8 Delays, GO1 Signal Table, Developed with Public Funds
26 APR 76	Version D	Supertypes GO1, GO2, GO3, Printouts Modified, GO1 Signal Table, Developed with Public Funds
NOV 76	Version E	---
11 JAN 77		Preliminary Fault Finder
3 MAY 77	GO	Fault Finder SYSFILE, FF1, FF2
30 NOV 77	GO	Types 16 & 17
3 MAR 78		Version D as documented for EPRI, Master Program GOFF, Data Decks Control Sequence, Alpha Descriptors, New Type 4
1 DEC 78	GOFF	Program FGO & GO4 Created
17 AUG 79	GO	Efficiency Update, New Program Structure, Procedures and CLISTS, LIBRARY GORUN
1 OCT 79	GO	Effect Evaluation EE1, EE2, EE3
20 MAR 80	GO	CDC Version Documented for EPRI
20 MAY 80	GO	IBM Version Documented for B&R, UP&L, EPRI
30 DEC 80	GO	IBM Version Enhanced at UCC, Dallas, Descriptors, Facility To Alter Array Sizes, Explanation Of Use
1 SEP 82	KSCGO	Version 1.0 Proprietary to KSC, Both VAX and CYBER Versions, Developed directly from GO Version C, 1976

Like most well-used software the GO codes have been continually changed and modified. In 1979-80 under the sponsorship of Utah Power and Light Company and EPRI the GO codes were converted from the Control Data Corporation Fortran, in which the most recent version had been written, to IBM Fortran. KSC accomplished this task and documented both the CDC and IBM versions of the codes as they existed in 1980. In 1982, KSC on its own initiative and funding developed a new proprietary version of the GO programs. Using the 1976 public domain Version C of GO, KSC scientists have created the latest version called KSC GO, Version 1.0, September 1, 1982. This latest version of GO is

available from KSC in both VAX-11 and CYBER versions. The VAX-11 version can be readily converted to IBM systems and has the added capability of word-packing to increase efficiency and reduce disk access requirements. This feature was not a part of the former IBM version of the code. An additional capability, which takes advantage of the large main memory of the VAX, is that array sizes can be increased to 20,000 or more (vs. the 3000-5000 array sizes on former versions) to effect orders of magnitude error reduction for large problems.

There are a number of other additional new features to KSC GO, Version 1.0. In particular the fault finder algorithm is completely new and is no longer tied to probabilities. These new features are fully documented in complementary publications.

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

LIST OF FIGURES

LIST OF FIGURES (CONTINUED)

LIST OF FIGURES (CONTINUED)

LIST OF FIGURES (CONTINUED)

CHAPTER 1
ASSESSMENT OBJECTIVES

INTRODUCTION

The objectives of a probabilistic system assessment must be defined before commencing the work. A number of factors must be considered to clearly define the objectives. In this chapter we discuss some of the factors which must be considered.

System Events and Their Probabilities of Occurrence

A fundamental question to be asked about a system is, "What do we want to know that we don't already know?" If the answer to that question is, "We want to know the probabilities of occurrence of certain system events," then it may be desirable to construct a system GO model to calculate these probabilities. If the system has a well-defined configuration and if pertinent component data can be obtained, then GO models can be developed to answer specific questions.

The KSC GO methodology is used to generate system events and their probabilities of occurrence from the defined interrelationships of external inputs and boundary conditions, constituent elements and components and the probabilities associated with their possible operational states. The methodology is used additionally to identify sets of component failures which singly or in combination cause system failures. Another principal use is to study the effects upon system performance caused by changes in or uncertainties about component probabilities of performance.

System Performance Measures

In the foregoing paragraphs the terms "system," "performance," and "data" have been purposely left vague. We now proceed to define them more carefully.

A system is any collection of basic elements (including human actions) which are arranged in a definite configuration and which interact to produce at least one output event of interest. This definition is broad. Examples of systems are household appliances, automobiles, industrial plants, power plants, missile systems, airplanes, satellites, communication networks, etc.

System performance is often measured using standard terms like reliability and availability. Definitions for these terms are:

> Reliability is "the characteristic of an item expressed by the probability that it will perform a required mission under stated conditions for a stated mission time" (IEEE Std. 577-1976).
>
> Availability is "the characteristic of an item expressed by the probability that it will be operational at a randomly selected future instant in time" (IEEE Std. 577-1976).

Safety and risk also are couched in terms of the probability of occurrence of catastrophic, hazardous or destructive events.

The objective of a system assessment is almost always the determination of the reliability, availability, safety, risk, or hazard associated with the design, fabrication, and use of a system. Because each of these measures is a probability, system performance is almost always expressed as the probability of occurrence of specific system events. The GO methodology was developed to calculate such probabilities.

Data

Different system performance measures require different types of component data. To generate system availability estimates requires estimates of component availabilities. To develop system reliability estimates requires component reliability estimates, etc.

In all cases, however, system GO models require component probability point estimates as contrasted with probability distributions. For example, consider a component which exhibits three mutually exclusive modes of operation, e.g., premature, normal or proper, and failure. Data for this component may establish that the probabilities of the component taking these operational states are 0.001, 0.997, and 0.002 respectively. These reliability point estimates are provided to the computer, along with similar data for other components to generate reliability point estimates for system events.

For an availability study, components will have only two states, i.e., available or unavailable. A given component may have an availability point estimate of 0.9999 and a corresponding unavailability point estimate of 0.0001 (1-0.9999). Using such data for all constituent model elements the GO software will generate system availability point estimates.

Probabilities associated with human actions and decisions are often included in GO models to account for such effects. These also are input in the form of point estimates.

Data for a GO model is a collection of pertinent probability point estimates that aptly describe the probabilistic behavior of the basic model elements. Often the level of resolution to which a GO model is developed depends upon the component data available.

Standard reference sources exist documenting generic data for most types of equipment, e.g., electrical components, pumps, valves, instrumentation. MIL-HDBK-217B, GIDEP, National Power Reactor Data System (NPRDS), and IEEE Std. 500-1977 Nuclear Reliability Data Manual are representative of available data sources.

Assumptions and Boundaries

All studies are limited by time and dollar constraints. To clarify the scope and intent of an analysis, the explicit and implicit assumptions made in developing a model, even though "obvious" or "universal," should be clearly stated. The fact that only a certain subsystem is being analyzed, that all electrical and human inputs are considered perfect, that no out-of-tolerance or beyond-limit stresses have been postulated, etc., should be clearly stated. This focuses attention on what is being analyzed and aides in interpreting and stating the study results.

Success and Failure Criteria

To develop a model which faithfully represents a system requires a clear definition of what constitutes both system success and system failure. Determining the criteria for success and failure usually simplifies the task of model development and sharpens the focus on the study objective.

Limitations and Uncertainties

Studies are limited in many ways. Time does not permit the accumulation of more data. The exact configuration is not known. The effect of certain component failures is not defined. Excluding analyst error, uncertainties result from lack of knowledge. Data uncertainties are often treated using data ranges, and expressing the results in terms of confidence intervals rather than precise point estimates. Configuration and component response uncertainties are treated by developing several models to handle the most likely situations and portray the differences caused by different configurations and component responses. The recognition and treatment of limitations and uncertainties will enhance the value of a study and will help clarify the assessment objective.

Design Enhancement

Having quantified system performance and taken into account knowledge deficiencies and uncertainties, an often fruitful approach expanding the study is to identify major causes of degradation or failure. Once identified these effects can often be mitigated with improved quality control or by design changes. The beneficial effects of such improvements or design changes can be objectively determined and reported by making the changes to the model and rerunning it. Consequently, the study objective may be to determine how a system can be improved to achieve a certain level of performance.

The scope of, and available resources for, the study, the schedule in which it must be completed, and the end product to be produced are additional considerations which help define the assessment objectives.

From a technical point of view the most important considerations in defining the study objective are the performance measures wanted, the system definition, the data required, and the assumptions and boundary conditions to be used.

It is difficult, if not impossible, to develop models and perform assessments when the study objectives are not clearly defined. As you read the remaining chapters of this primer and become familiar with the KSC GO methodology, focus on the objectives for which various example GO models are developed. Doing so will help you catch the essence of the procedure and illuminate the software capabilities. Defining the objectives to be achieved by developing GO system models will aid you in applying the procedure to real world problems of your own.

CHAPTER 2
EVENT TREES

Introduction

Every person who has engaged in problem solving exercises has employed branching diagrams or event trees. The purpose of this chapter is to illustrate how event trees are used to solve reliability problems. This discussion is a preamble to using the GO methodology. The GO computer programs automate the process of creating and manipulating event trees. Their efficient use permits the creation and manipulation of event trees of almost any size.

The Coin Flip Problem

Let us presume we flip a coin four times. Five possible outcomes result, i.e., 0, 1, 2, 3, or 4 heads. We desire to determine the probabilities of obtaining each of these elemental outcomes.

We can formalize the procedure by defining a random variable X to be the number of heads obtained in the experiment. As noted above, the random variable X can take any of five different values. The problem then is to determine the probability that X takes each specific value, i.e., $Pr(X=i) = ?$, $i = 0,1,2,3,4$. In summary we desire to complete the following figure (Figure 1).

VALUE X TAKES	PROBABILITY
0	?
1	?
2	?
3	?
4	?

FIGURE 1 PROBABILITY MASS FUNCTION FOR RANDOM VARIABLE X

This problem can be readily conceptualized with an event tree (see Figure 2). Each flipping operation doubles the branches in the tree. Ultimately the tree has 16 mutually exclusive terminal branches.

The event tree is converted to a probability tree by introducing the probabilities associated with each of the two possible outcomes for each of the four independent experiments. We assume there is an equally likely chance that either a head or a tail outcome occurs on each flip. Consequently each has a 0.5 probability of occurring. Because each outcome is equally likely each of the sixteen terminal branches have an equal likelihood of occurring, namely $(.5)^4 = 0.0625$.

Note on Figure 2 that only one terminal branch produces four heads. Consequently, the probability of obtaining four heads is 0.0625. The same is

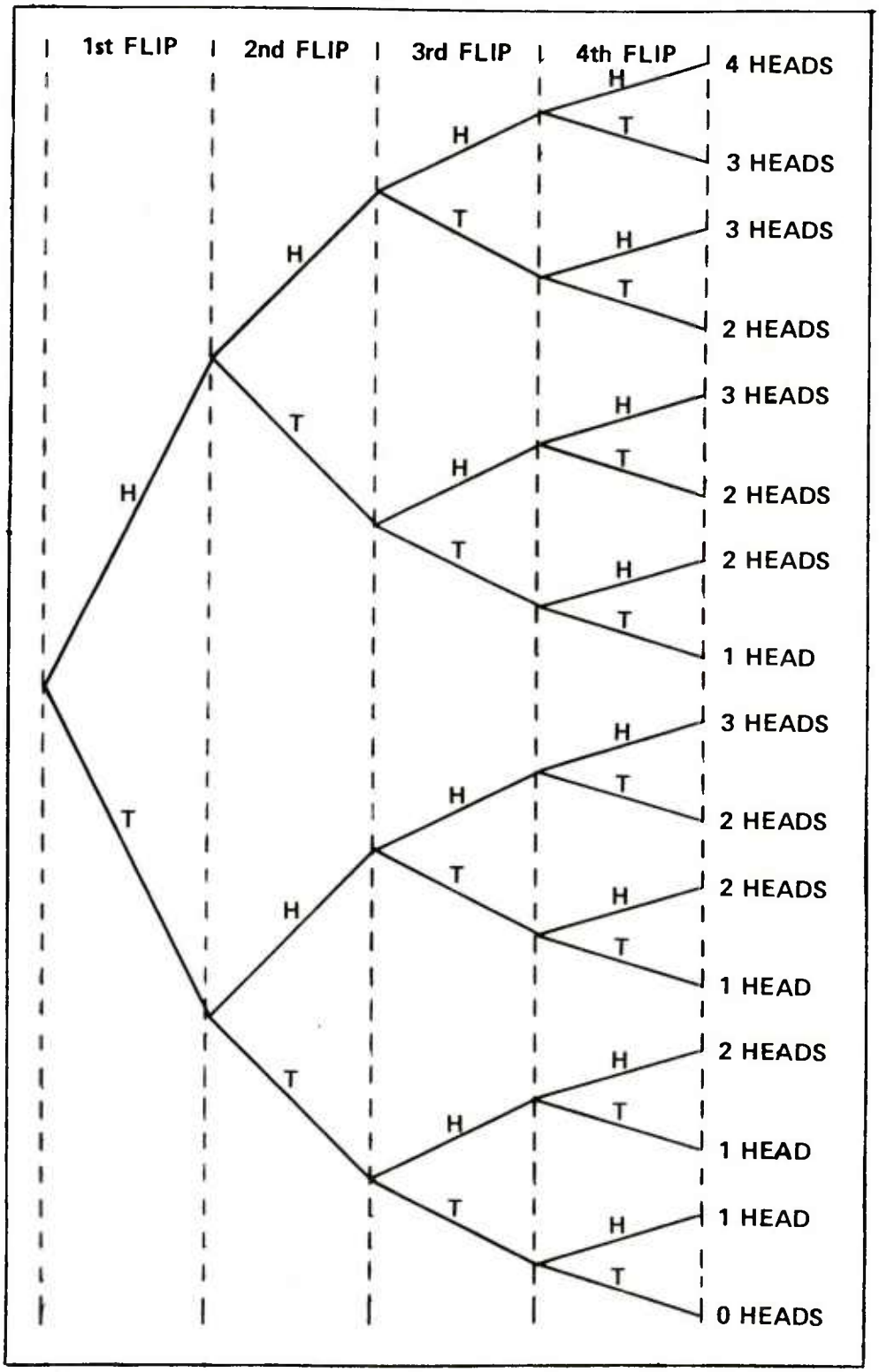


FIGURE 2 EVENT TREE FOR COIN FLIP PROBLEM

true for zero heads. There are four terminal branches resulting in three heads, so the probability of this outcome is 4x0.0625=0.25. Similarly there are four terminal branches producing one head. Finally there are six terminal branches producing two heads with probability of occurrence 6x0.0625=0.3750.

The table for the probability mass function for X can now be completed from the event tree solution* to this problem.

VALUE X TAKES	PROBABILITY
0	0.0625
1	0.2500
2	0.3750
3	0.2500
4	0.0625
TOTAL	1.0000

FIGURE 3 PROBABILITY MASS FUNCTION FOR RANDOM VARIABLE X

Switch Operation Analysis By Event Tree

We move now to a less obvious analysis using the event tree approach. Consider the operation of a simple switch as shown below:

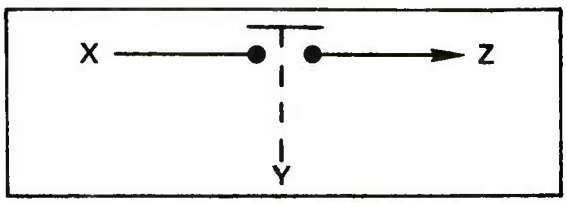


FIGURE 4 SWITCH

The main electrical input to the switch is shown as random variable X, which takes possible values 0, 1, and ∞ representing the times of arrival of input power to the switch. The mechanical actuation of the switch is shown as random variable Y, which similarly takes values of 1 and ∞ indicating the times at which the switch is actuated. Given that the probability mass functions for both X and Y are known, find the probability mass function for dependent random variable Z, whose values represent the times at which output power is available from the switch.

---

* For this trivial problem there are a number of solution methods. Perhaps the most straightforward is the binomial expansion of $(H + T)^4 = H^4 + 4H^3T + 6H^2T^2 + 4HT^3 + T^4$ where H is the probability of obtaining a head and the exponent on H is the number of heads in that term. The same nomenclature is used for tails. Since H=T=1/2 in this problem the result is immediate.

First we define the following events:

X_i = Event that power is applied to switch at relative time i.

Y_i = Event that the switch is actuated at relative time i.

Z_i = Event that power is received from the switch at relative time i.

The probability mass functions for random variables X and Y are defined in the figure below where the possible times (or values the random variables take) are 0, 1, and ∞. In this case infinity means 'never', i.e., power never arrives or the switch is never actuated, zero means before the intended time of operation, and 1 means the time of intended operation (power arrival and switch actuation).

i	$P(X_i)$	i	$P(Y_i)$
0	.1	1	.8
1	.5	∞	.2
∞	.4		

FIGURE 5 PROBABILITY MASS FUNCTIONS FOR DISCRETE RANDOM VARIABLES X AND Y

If the switch itself is perfect, the event tree of Figure 6 shows the solution to this problem. Six unique joint events result from the combination of the two independent random variables X and Y taking their respective values. The resultant values assumed by random variable Z are 1 and ∞, meaning that the switch provides electrical power at time 1 with probability 0.48 and that it fails to provide an output (i.e., provides an output at time ∞ or never) with probability 0.52.

Now, instead of presuming that the switch is perfect, let us presume that it has three operational states -- good (g), failed (f), and premature (p) and that it takes on these states with probabilities 0.7, 0.2, and 0.1, respectively. When the probabilistic behavior of the switch itself is additionally considered, the six joint events resulting from the times of arrival of input power (random variable X) and the time of actuation (random variable Y) expand to 18 joint events.

The event tree for the case of an imperfect switch is shown in Figure 7. Note that now there are three possible outcomes for random variable Z. It can take values 0, 1, and ∞. In this case a value of 0 means a premature electrical output from the switch which can only occur if input power is premature (X_0) and the switch is prematurely electrically continuous (shorted). Now the output distribution for Z is:

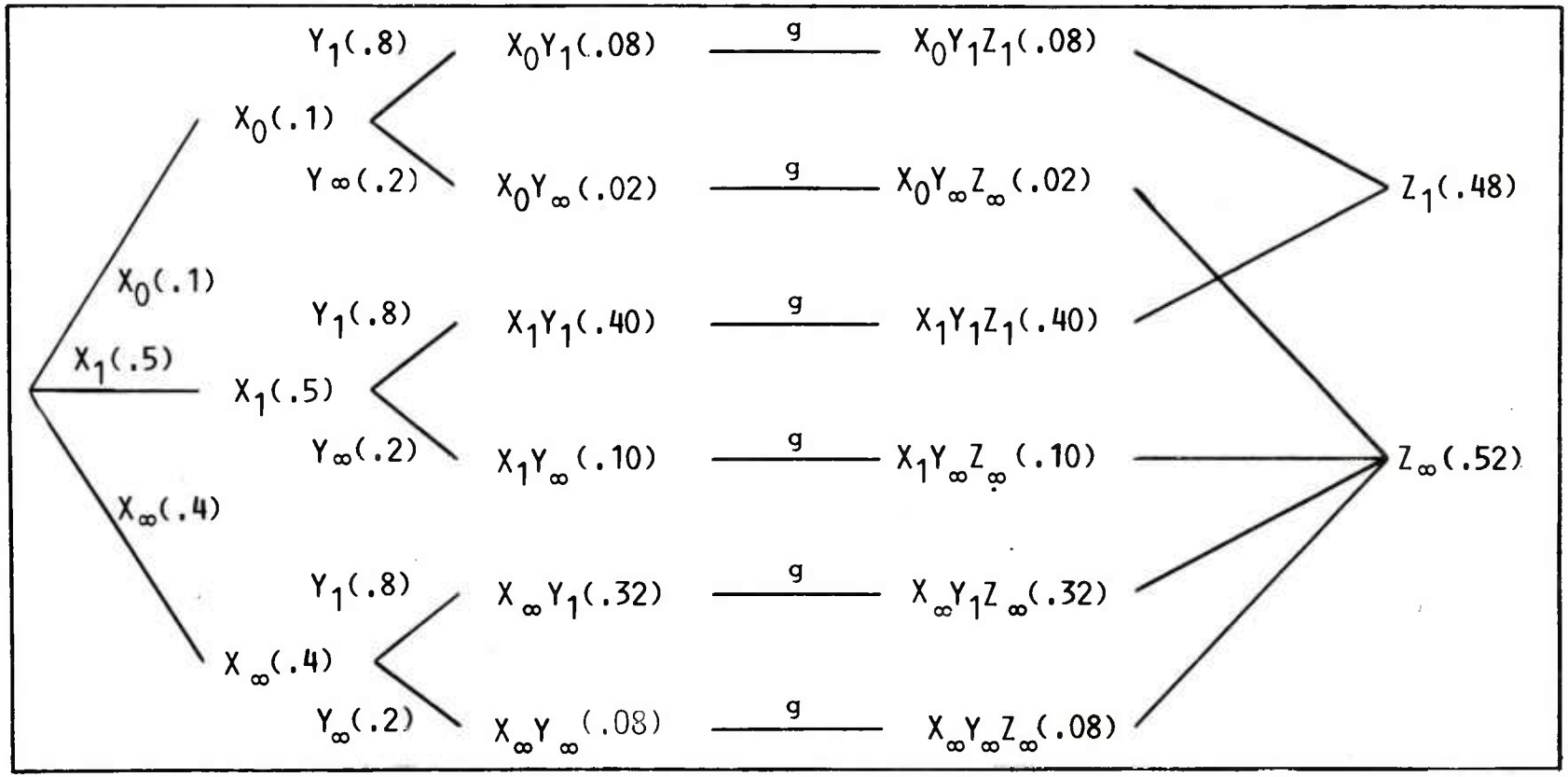


FIGURE 6 EVENT TREE FOR PERFECT SWITCH

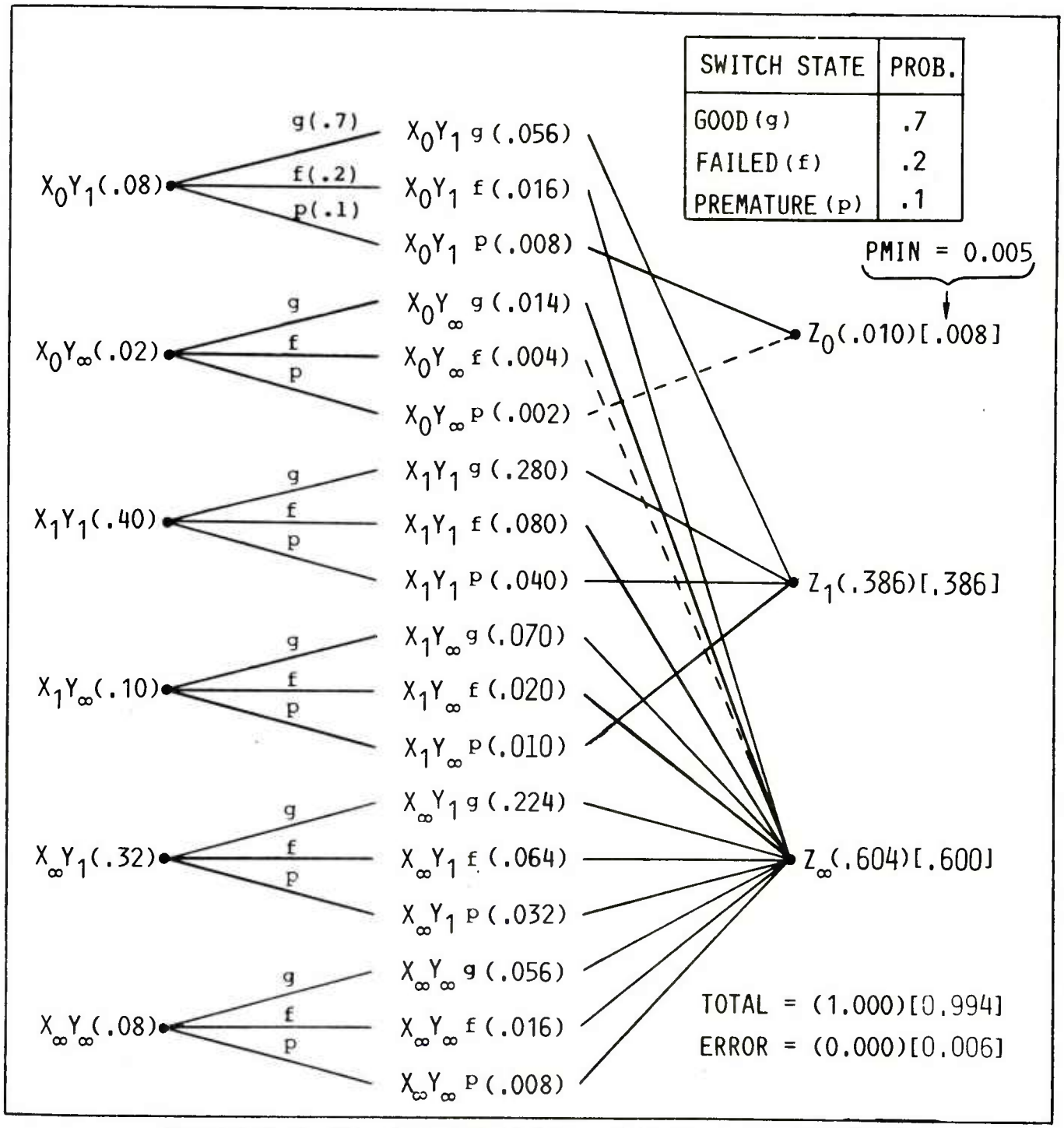


FIGURE 7 EVENT TREE FOR IMPERFECT SWITCH

$P(Z_0) = 0.010,$
$P(Z_1) = 0.386,$
$P(Z_\infty) = 0.604.$

For this relatively trivial problem, 18 possible unique joint events involving the three random variables (X, Y and the switch) resulted. For a system of many components the event trees become very large. Consequently, to keep the trees manageable in size, terms with smallest probabilities of occurrence are often discarded. This is done by specifying a probability value (PMIN) and discarding (pruning) all terms (branch) combinations whose probabilities of occurrence are less than this preassigned value. Even if the event trees are generated automatically using a computer it is generally necessary to prune the trees if the number of system components is more than a few dozen.

In the event tree for the imperfect switch (Figure 7), the branches with dashed lines leading to the Z events would have been eliminated with a PMIN of 5×10^{-3}. The probabilities of occurrence of final Z events are reduced by the probabilities of the discarded terms and some error is introduced. In this problem note that a total error of 6×10^{-3} is introduced with a PMIN of 5×10^{-3}.

Even though this is undesirable, when large systems with hundreds of components are analyzed, the introduction of some relatively small error is a proper tradeoff for having a tractable problem. Experience has shown that with some skill in structuring the trees, the error can be made insignificantly small in almost all cases, while permitting economical computer manipulation of very large trees.

CHAPTER 3
THE GO METHODOLOGY

INTRODUCTION

The GO methodology is an automated way to develop system event tree models. Such models are developed to analyze the probabilistic behavior of systems defined by such terms as reliability, availability, safety and risk. GO models are developed using standard logical operators to represent system equipment and components, their operations and interactions. Appropriate probability data for equipment performance, human actions, and external events are gathered and tabulated. A GO model and its associated probability data are then placed on computer data input files which are accessed and processed by the GO computer programs to develop the event trees whose terminal branches contain the desired system events and their probabilities of occurrence.

GO Operators (Types)

The heart of the GO process is the definition of fifteen standard logical operator "types" which are used to represent model elements and interactions. For example, a type 1 operator represents the logical operation of an equipment which either performs, or fails to perform, its function given a proper input or stimulus. The type 2 operator performs the logical OR gate operation where a successful response is generated if any of several inputs is proper, etc.

The fifteen logical operators, for which algorithms are defined in the GO codes, are depicted in Figure 8. Figure 8 also shows the symbol normally used to define the operators. Operator inputs are called stimuli (S_1, S_2, ... S_n), and outputs are called responses (R_1, R_2, ..., R_n). Such inputs and outputs are random variables. An operator, which represents equipment responses or human actions, and which may itself have associated performance probabilities, processes the input random variables in a prescribed and well-defined way to generate the output random variables. In GO models these random variables are called "signals", a carryover from electrical terminology.

GO Models

A GO model is developed directly from electrical schematics, engineering blueprints, and flow diagrams by substituting the standardized GO operators for physical equipment, human actions and logical operations. The inputs and outputs to these operators are combined and arranged to show the logical

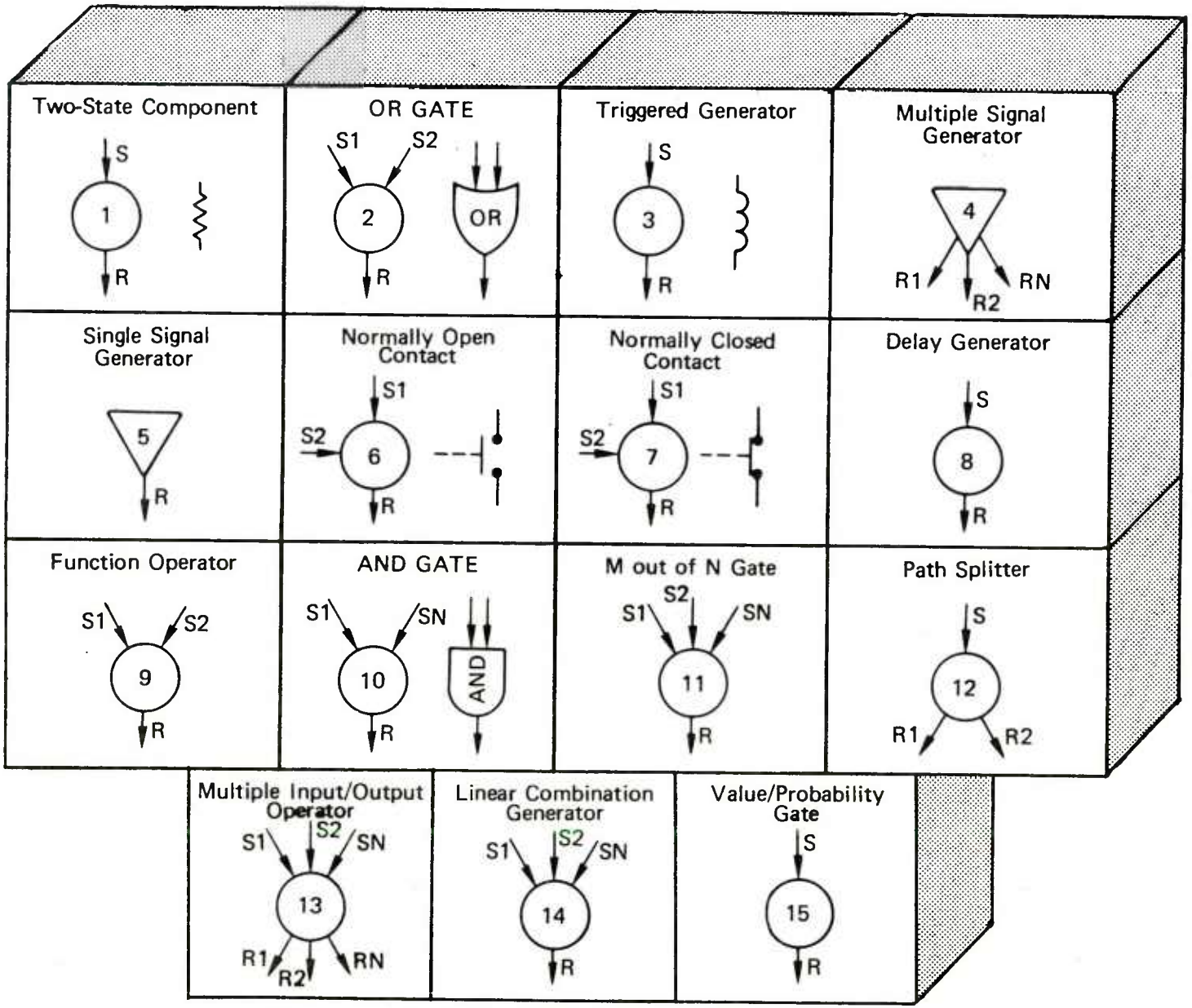


FIGURE 8 GO OPERATOR TYPES

configuration or process flow necessary for successful operation of the systems. The GO models thus appear similar to the original schematic, blueprint or flow diagram and capitalize upon the analyst's familiarity with the configuration and the proper intended operation of the system. This similarity to such diagrams aids in validating, understanding and interpreting the models.

The steps in creating a GO model are: (1) learn how the system is configured and actually operates; (2) define system success and failure criteria; (3) identify the system events about which information is sought; (4) represent system elements with standardized GO operators; (5) combine the inputs and outputs of operators representing system elements into a GO model portraying successful system operation; (6) obtain the probabilistic data for component responses.

To illustrate how this is done, consider the example system of Figure 9. An electrical cord is plugged into a wall circuit. Subsequently Sam turns on the switch to a motor. Question: What is the probability that the motor begins operating when Sam turns on the switch? The answer to this question is a function of the availability of power, the electrical continuity of the plug and cord, the reliability of the switch, whether or not Sam actually turns the switch on, and the reliability of the motor.

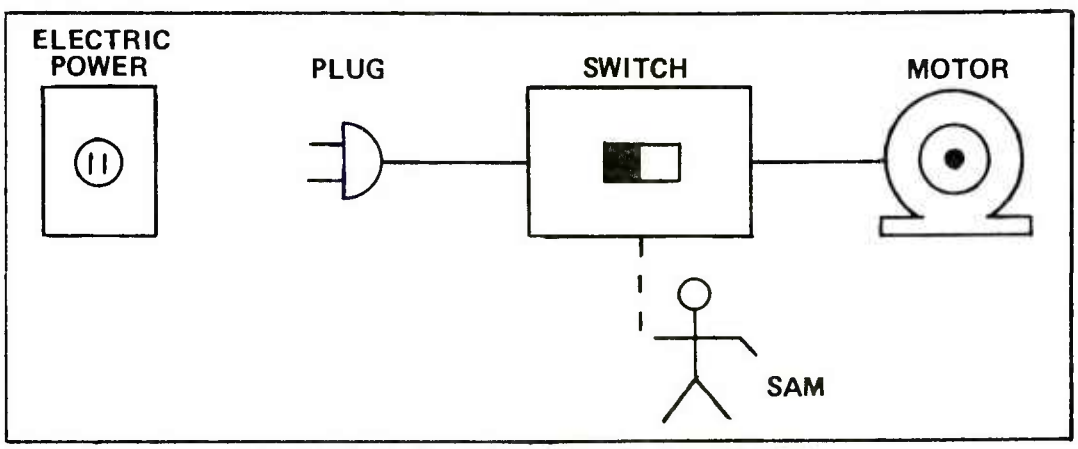


FIGURE 9 EXAMPLE SYSTEM

A GO diagram for this system is shown in Figure 10. The symbols for the standard GO operator types are used to represent each system element. Inside each symbol is a hyphenated number. In each case the first number specifies the operator type. The second number, called the "kind" number, references the probability data associated with this operator. It will be addressed in the next section.

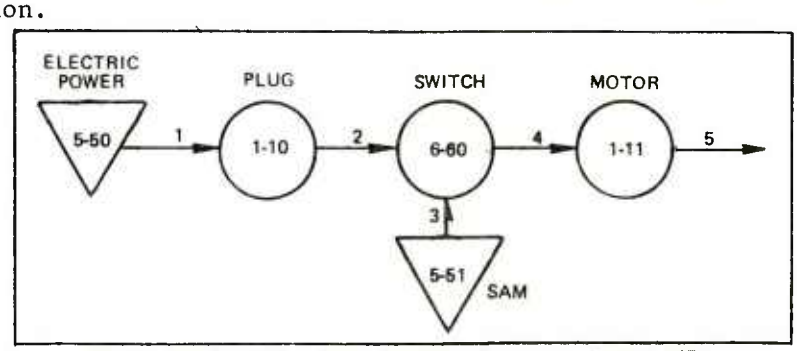


FIGURE 10 EXAMPLE GO MODEL

Each operator input and output is given a "signal" (random variable) number. For example the availability of power is represented in this model by signal #1, the output from the type 5 operator representing the availability of electric power. The output from the type 1 operator representing the plug is signal #2, etc. Signal #5, the output from the type 1 operator representing the motor, is the final signal generated by the model. In fact, the model was developed to generate the probability mass function for signal #5 to allow us to quantify the probability of motor operation in this system.

GO1 Operator Data

Figure 11 shows the format for communicating the operator type, the associated kind data reference, and the input and output signal numbers to computer program GO1 as shown in Figure 11. To record the operator data for a specific component, the operator type is written first, next the kind number (K) is generally written, then the input signal number or numbers (S_i) are written followed by the output signal number or numbers (R_i). For example the information concerning the plug in the GO model of the example system (Figure 10) is communicated to the computer in the form,

```
1  10  1  2  $  PLUG
```

This tells the computer that we have a type 1 operator with kind 10 probabilities. It has signal #1 as an input and generates signal #2 as an output. The dollar sign ($)* is a data terminator and the description to the right of the terminator is optional. Obviously a signal cannot be used as an input until it has been created as an output by some prior operator.

The entire set of GO1 data for the example system is recorded in the following eight lines:

```
GO1 DATA FOR EXAMPLE SYSTEM
 $PARAM INFIN=3 $
5 50 1 $ ELECTRIC POWER
1 10 1 2 $ PLUG & CORD
5 51 3 $ SAM
6 60 2 3 4 $ SWITCH
1 11 4 5 $ MOTOR
0 5 $ FINAL OUTPUT SIGNAL
```

The first line of data is a model name or header card. The second line of data calls up a Fortran namelist parameter named PARAM which permits the initialization of several parameters. In this model only the parameter INFIN, meaning the largest value a random variable may take, is specified. Default values are used for several other GO1 parameters which can be defined, e.g., BIAS, OP, SIGNAL. (See the GO Reference Manual for full exposition of parameters).

* Some versions of GO use a slash (/) as a terminator.

1 K S R

2 0 n S_1...S_n R

3 K S R

4 K n R_1...R_n

5 K R

6 K S_1 S_2 R

7 K S_1 S_2 R

8 K S R

9 K S_1 S_2 R

10 0 n S_1...S_n R

11 m n S_1...S_n R

12 K S m R_1...R_m

13 K n S_1...S_n m R_1...R_n

14 K n S_1...S_n R

15 K S R

K: KIND NUMBER

S: STIMULUS (INPUT) SIGNAL NUMBER

R: RESPONSE (OUTPUT) SIGNAL NUMBER

SEE THE GO REFERENCE MANUAL FOR COMPLETE DESCRIPTION OF INPUT DATA

FIGURE 11 GO1 OPERATOR DATA

After the header and parameter cards, the operator data is entered, one operator per line. The G01 operator data completely defines the logical nature of the model elements and their interactions. The G01 data thus documents the system configuration.

In this model there are five operators. The first operator, which represents the availability of electric power, is a type 5 operator. It references kind 50 probabilities and generates signal #1. Then, in succession, we have a type 1 kind 10 operator representing the plug and cord, a type 5 kind 51 operator representing Sam, a type 6 kind 60 operator representing the switch, and a type 1 kind 11 operator representing the motor.

The output from the motor is signal #5. This is the final signal whose distribution we desire to see in the computer results. We tell the G01 software that signal #5 is the signal to appear in the output with the last line of entry. The use of a "0", instead of an operator type, flags the computer that this entry terminates the G01 data and that the following entries on that line are signals which are to appear in the final output distribution. In this case only the single signal, #5, is requested.

GO Values and Event Definitions

An important consideration in developing GO models, not obvious in the discussion to this point, is the specification and meaning of the possible values that the random variables may take. Recall that in the coin flip example of the random variable, number of heads obtained in four flips, could take any of the five possible values - 0, 1, 2, 3, or 4.

Similarly in every GO model the possible values which the random variables (signals) may take must be specified. The model is clarified by attaching a meaning and definition to each value.

For example if one desires to know whether a system functions or fails to function, two values would be sufficient. In such a GO model these would be the values 0 and 1. The value 0 would mean success and the value 1, failure.

To pursue this concept further, let us define some values for the GO model of the example system of Figures 9 and 10. For the purpose of illustration let there be four possible values with definitions as shown in Figure 12.

```
0 - All prior time
1 - Time when electric power is available in wall socket
2 - Time when Sam turns on switch
3 - Never
```

FIGURE 12 VALUES FOR EXAMPLE SYSTEM

With these definitions we see that the random variables will take values which reference a relative time sequence. Having defined these values for this model, the meaning of random variable #5 taking value 2, 5_2*, means the event that the motor turns on when Sam turns on the switch. Similarly, event 5_3, means the motor never turns on, or fails to turn on, etc.

For the example system some additional events which may be defined are listed in Figure 13.

EVENT	EVENT DEFINITION
1_1	POWER IS AVAILABLE TO THE PLUG AT TIME 1
2_1	POWER IS AVAILABLE FROM THE PLUG AT TIME 1
3_2	SAM OPERATES SWITCH AT TIME 2
4_2	POWER IS AVAILABLE FROM THE SWITCH AT TIME 2
4_3	POWER IS NEVER AVAILABLE FROM THE SWITCH
5_2	THE MOTOR OPERATES WHEN SAM TURNS ON THE SWITCH
5_3	THE MOTOR FAILS TO OPERATE

FIGURE 13 SOME EVENTS DEFINED FOR EXAMPLE PROBLEM

GO2 Probability Data

The probability data required for a GO model is specific to that model. It is entered by specifying an arbitrary kind number which permits reference to the data. The form in which the data is entered is determined by the type of operator which calls for the data. Figure 14 depicts the format in which the probability data for each operator type will be entered.

The arbitrary kind numbers are represented by K in the figure. The probabilities of good, bad, and premature component operation are represented as P_g, P_b, and P_p respectively. Values are represented with V.

The probability data associated with a type 1 operator is provided to computer program GO2 in the form:

K 1 P_g P_b

* We adopt the nomenclature that events are defined by signal numbers subscripted with specific values.

K 1 P_g P_b

K 3 P_g P_b P_p

K 4 n m $V_{11} \cdots V_{1n}$ P_1
$V_{21} \cdots V_{2n}$ P_2
$\vdots$
$V_{m1} \cdots V_{mn}$ P_m

K 5 n V_1 $P_1 \cdots V_n$ P_n

K 6 P_g P_b P_p

K 7 P_g P_b P_p

K 8 n D_1 $P_1 \cdots D_n$ P_n

K 9 n X_1 $Y_1 \cdots X_n$ Y_n

K 12 m $P_1 \cdots P_m$

K 13 n m N
$VS_{11} \cdots VS_{n1}$ M_1
$VR_1 \cdots VR_m$ P_{11}
$\vdots$
$VR_1 \cdots VR_m$ $P_{M_1 1}$
$\vdots$
$VS_{1N} \cdots VS_{nN}$ M_N
$VR_1 \cdots VR_m$ P_{1N}
$\vdots$
$VR_1 \cdots VR_m$ $P_{M_n N}$

n : Number of inputs; $0 \leq n \leq 10$

m : Number of outputs; $1 \leq m \leq 10$

N : Number of output sets;
(if n=0, N=1); $1 \leq N \leq 10$

M_i : Number of output terms for the i^{th} output set; $1 \leq M_i \leq 10$

K 14 n $a_1 \cdots a_n$ a_o

K 15 V_1 V_2 V_3 V_4 P_1 P_2

K: KIND NUMBER
P: PROBABILITY
V: VALUE
D: DELAY

SEE THE GO REFERENCE MANUAL FOR COMPLETE DESCRIPTION OF INPUT DATA

FIGURE 14 GO2 KIND DATA

If we refer to the GO model of the example system (Figure 10) the probability data for the plug, shown as a type 1 kind 10 operator in Figure 10, might be as follows:

```
10  1  0.9  0.1  $  PLUG
```

This indicates that kind 10 is referenced by a type 1 operator. That operator has a success probability of 0.9 and a failure probability of 0.1. The dollar sign ($) is again a data terminator and optional descriptive information about the data can be placed to the right of the terminator.

The probability data needed for a specific GO model must be developed by the analyst. This will usually be done by referencing a data base or by engineering judgement.

The analyst must insure that the data being used is appropriate. For example there are three types of data which are often used for system analyses. These are (1) per demand probability estimates, (2) reliability with time estimates, and (3) availability estimates.

Per demand or startup estimates are derived from binomial trials. Given, from experience, that there have been x successful starts in n trials the reliability point estimate, R, for successful startup is given as the expected value $\frac{x}{n}$, hence $R_i = \frac{x_i}{n_i}$ for the i^{th} component. An unbiased estimate for the variance, V, of the reliability is also provided by the expressions,

$$V = \frac{\left(\frac{x}{n}\right)\left(1 - \frac{x}{n}\right)}{n} = \frac{x(n-x)}{n^3} .$$

This variance estimate permits the treatment of uncertainty in system estimates as a function of uncertainties in component estimates, but is, of course, invalid if $x=0$ or $x=n$. If either of these cases occur arbitrary variance estimates must be generated by some other means.

The reliability with time estimates are generated using available failure rate data. Using λ_i as the failure rate for the i^{th} component, the reliability, R_i, of that component at time t is,

$$R_i = e^{-\lambda_i t} .$$

This is the probability that the component has not failed since being placed in service up to time t. The formulation presumes, that the failure rates are constant and therefore that the time-to-failure distributions are exponential. (Other time-to-failure distributions could be used, e.g., uniform, lognormal, normal, gamma, Weibull, etc.)

Availability estimates for a system can be generated from component availability point estimates using GO models. The system availability estimate, A, is a function of the component availability estimates. These are denoted as A_i for the i^{th} component. Hence,

$$A = f(A_1, A_2, \ldots, A_n).$$

The component availability point estimates can be derived using the well established function,

$$A_i = \frac{MTTF_i}{MTTF_i + MTTR_i},$$

where,

A_i = Availability point estimate of the i^{th} component,
$MTTF_i$ = Mean-Time-To-Failure of the i^{th} component,
$MTTR_i$ = Mean-Time-To-Repair of the i^{th} component.

If the times-to-failures are exponentially distributed, then $MTTF_i = \frac{1}{\lambda_i}$ where λ_i is the failure rate of the i^{th} component. Since several sources exist containing published generic failure rate data for most components, representative estimates for $MTTF_i$ are available for most types of components. Less data is available to establish representative estimates for the MTTR. These estimates are functions of the general maintenance philosophy - frequency of test, training, stock levels, etc. Where component MTTF and MTTR data are not available for modeled components, engineering estimates can be made of their availabilities.

The actual GO2 data entered for the example system is recorded in the following seven lines.

```
GO2 DATA FOR EXAMPLE SYSTEM
 $PARAM $
10 1 0.9 0.1 $ PLUG
11 1 0.7 0.3 $ MOTOR
50 5 1 1 1. $ ELECTRIC POWER
51 5 1 2 1. $ SAM
60 6 0.8 0.2 0.0 $ SWITCH
```

As in program GO1, the data for program GO2 commences with a model name or header line entry. The second entry is again a Fortran namelist parameter called PARAM which permits the setting of the parameter PERFECT. In this case the parameter PERFECT was left at its default value.

The next five lines of data record the kind numbers associated operator types, and the probability data for each component. The first two entries on each line are the kind and type. The remaining entries are the required

probabilities and other data necessary to define the probabilistic operation of a particular operator type. (Refer to Figure 14 which shows the format of the data entries for each operator type.)

In this model kinds 10 and 11 are referenced by type 1 operators, kinds 50 and 51 by type 5 operators, and kind 60 by a type 6 operator. The GO2 data for the type 1 operators records the success and failure probabilities of operation. The GO2 data for the type 5 operators indicates that in each case there is only one value which is taken with certainty. Electric power is available at time 1 with certainty, and Sam throws the switch at time 2 with certainty. The GO2 data for kind 60 type 6 contains the success, failure, and premature probabilities of operation. In this case the premature probability is specified to be 0.0.

GO Execution

To execute the GO sequence three computer programs in the set of GO codes are called upon to process the model data. These are programs GO1, GO2 and GO3. Program GO1 processes the operator data which defines the system structure. Program GO2 processes the probability data insuring its internal consistency and proper reference from operators processed in GO1. Program GO3 executes the operators in the GO1 data and introduces the probability data from GO2 to generate the event tree for the specific model being processed.

To portray how this is done representative data for the example system of Figures 9 and 10 have been recorded in Figure 15.

Note that the data is separated into three sets -- one for each of the three GO programs to be executed, GO1, GO2, and GO3. We have previously discussed the data for programs GO1 and GO2. In each set there is a name or header card, and a parameter card. The parameter card for GO1 defines INFIN=3 which tells the computer that the largest value which the random variables may take in this analysis is 3. Consequently the permissible values will be 0, 1, 2, and 3, with the associated meanings previously defined. The subsequent data in GO1 defines the operators and associates their inputs and outputs. The final card commencing with a 0 entry flags the computer that this is the end of the GO1 data and that signal 5 is to be the output signal whose mass density function is to be displayed.

The GO2 data reflects the probabilities which were assigned for this analysis. Note that both the availability of electric power and Sam's action are perfect but that they occur at time 1 and time 2 respectively. That is, power is certain to be available at time 1 and Sam is certain to actuate the switch at time 2.

In GO3 the parameter PMIN which could permit truncation of low probability events is set to 0.0. Consequently, no pruning will occur.

```
GO1 DATA FOR EXAMPLE SYSTEM
 $PARAM INFIN=3 $
5 50 1 $ ELECTRIC POWER
1 10 1 2 $ PLUG & CORD
5 51 3 $ SAM
6 60 2 3 4 $ SWITCH
1 11 4 5 $ MOTOR
0 5 $ FINAL OUTPUT SIGNAL
EOR*
GO2 DATA FOR EXAMPLE SYSTEM
 $PARAM $
10 1 0.9 0.1 $ PLUG
11 1 0.7 0.3 $ MOTOR
50 5 1 1 1. $ ELECTRIC POWER
51 5 1 2 1. $ SAM
60 6 0.8 0.2 0.0 $ SWITCH
EOR
GO3 DATA FOR EXAMPLE SYSTEM
 $PARAM PMIN=0.0 $
EOR
EOF
```

FIGURE 15 GO DATA FOR EXAMPLE SYSTEM

An event tree for this example system can be easily created. Such a tree is shown in Figure 16. Moving from left to right the branches in the tree are depicted in the order in which the operators were introduced in the GO1 data. The plug, switch, and motor states were represented by P, S, and M respectively subscripted with g, for good, and f, for failed. Events are also shown using the subscripted signal numbers.

Note that only one branch of the tree leads to a system success. This is event 5_2 defined as successful motor operation when Sam turns on the switch. All other branches lead to motor failure because of plug switch and motor failure.

The event tree which is actually created by GO3 is slightly different than that of Figure 16. The difference is that after each operator is processed all identical events are combined. This reduces the number of branches and results in more efficient processing. Figure 17 portrays how this branch combination process is employed for the example system to generate the two final output events, 5_2 and 5_3.

If more visibility about other combinations of the signals is desired we could specify them as final output signals. If every signal were specified as a final output signal the tree of Figure 18 would be generated. In this case there are four final events, representing the four unique combinations of the random variables. As before identical events generated are combined reducing the number of branches in the tree.

* EOR and EOF mean End-of-Record and End-of-File respectively. Different computers use different nomenclature for these expressions.

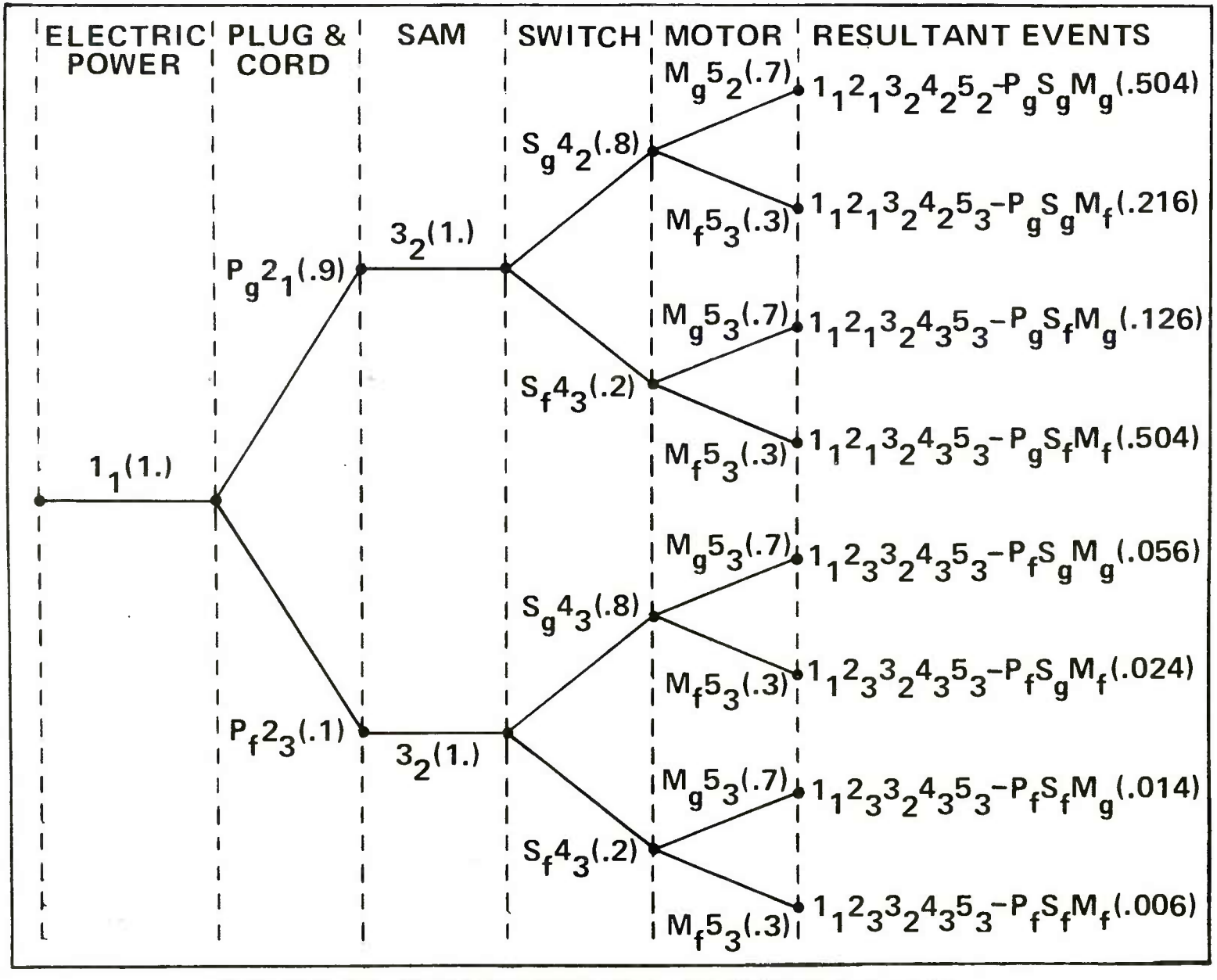


FIGURE 16 EVENT TREE FOR EXAMPLE SYSTEM

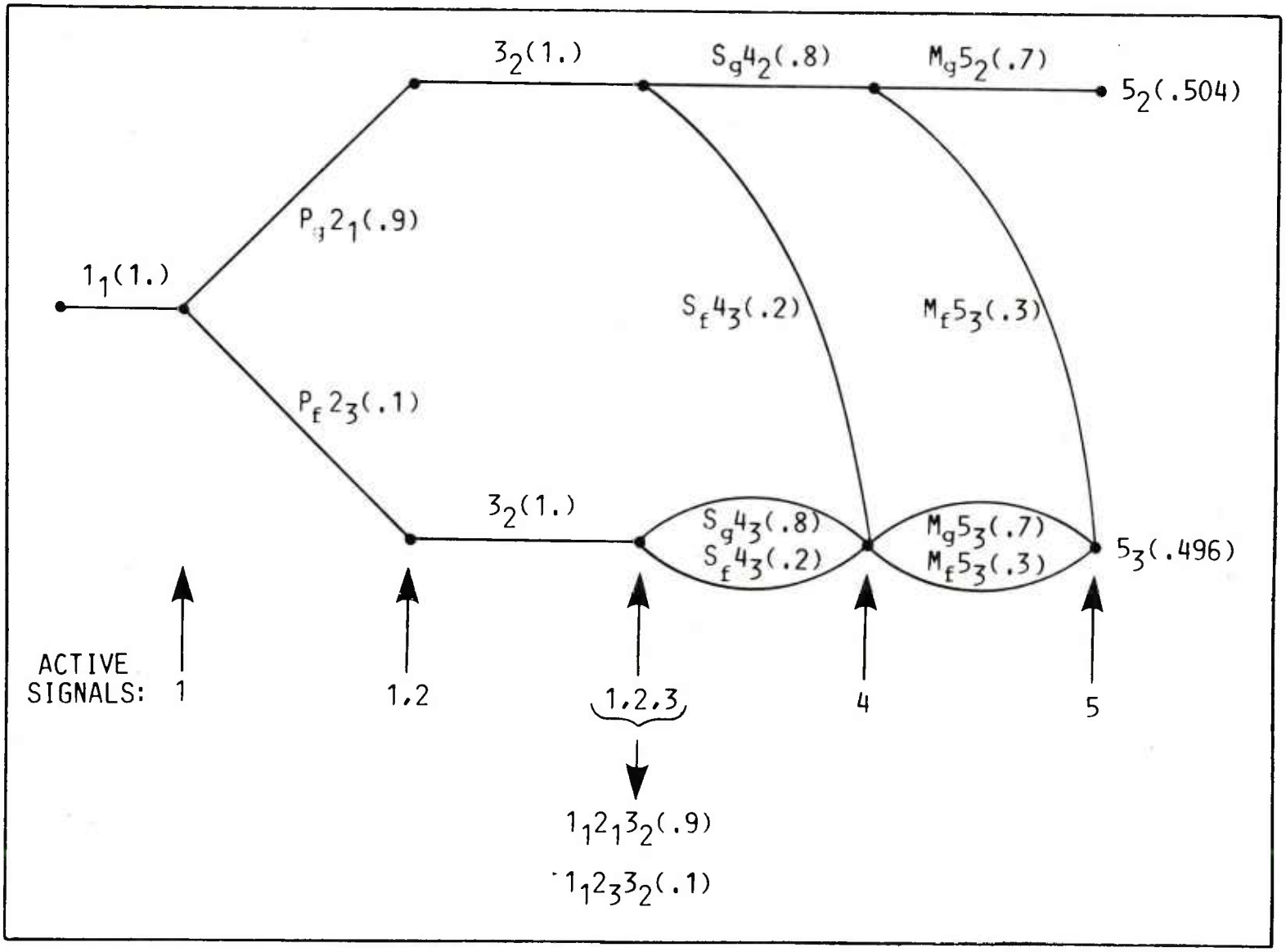


FIGURE 17 GO EVENT TREE FOR
EXAMPLE SYSTEM WITH SIGNAL 5 AS THE ONLY OUTPUT

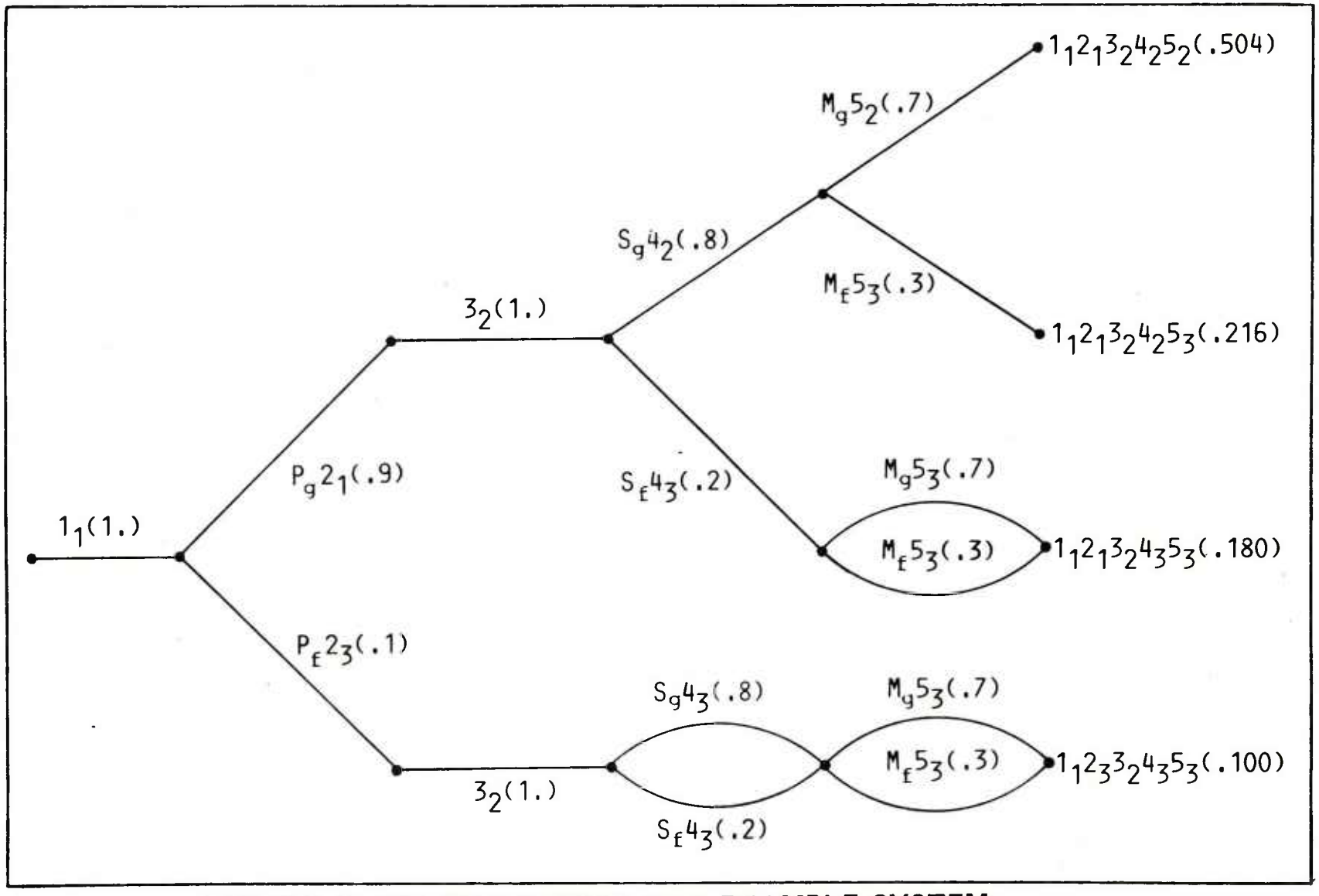


FIGURE 18 GO EVENT TREE FOR EXAMPLE SYSTEM WITH ALL SIGNALS AS OUTPUTS

GO Results

In most GO analyses the event trees are not depicted. Usually, only the final branches of the tree are retained and recorded because they provide the information about system performance which the model was created to generate. Internal information, while optionally available, is almost always suppressed to eliminate voluminous output.

The results of a GO analysis include the list of all possible system event states and their probabilities of occurrence. The GO3 output for the example system previously discussed is shown in Figure 19.

In Figure 19 the final event table records the resultant branches of the event tree of Figure 17. Signal 5 was the only output signal specified. As shown in Figure 19, signal 5 can take two

```
FINAL EVENT TABLE (INFINITY =   3)
                SIGNALS AND THEIR VALUES
                -------------------------
PROBABILITY       5
------------   ----
0.4960000000      3
0.5040000000      2
-------------------------------------
TOTAL PROBABILITY = 1.0000000000
TOTAL ERROR =         .0000000000
```

FIGURE 19 GO3 RESULTS FOR EXAMPLE SYSTEM

values, 2 and 3. The probability that signal 5 takes value 2 (the system success event) is 0.504. The probability that signal 5 takes value 3 (the system failure event) is 0.496 for this system. Note also that because no terms were discarded (no branches pruned) there is zero calculational error.

A second execution of this model was made specifying all of the random variables as final output signals. This was done by altering the last GO1 data entry to read:

```
0 1 2 3 4 5 $ FINAL OUTPUT SIGNALS
```

The results from this run are shown in Figure 20. We have now generated the complete joint distribution of the five random variables generating four unique events. These are: $1_1 2_3 3_2 4_3 5_3$, $1_1 2_1 3_2 4_3 5_3$, $1_1 2_1 3_2 4_2 5_3$, and $1_1 2_1 3_2 4_2 5_2$.

In addition to the complete joint distribution generated by the computer program, the code also produces the marginal distribution for each individual signal. (A marginal distribution is outlined by adding the probabilities of occurrence of a single signal taking a specific value without regard to the

values taken by other signals in the joint events.) Consequently, as in the prior case, we see that signal 5 can take only the values 2 and 3 with the probabilities previously noted. Also note that signals 1 and 3 only take a specific value with certainty because this is the way we specified the input data.

FINAL EVENT TABLE (INFINITY = 3)

SIGNALS AND THEIR VALUES

PROBABILITY	1	2	3	4	5
.1000000000	1	3	2	3	3
.1800000000	1	1	2	3	3
.2160000000	1	1	2	2	3
.5040000000	1	1	2	2	2

TOTAL PROBABILITY = 1.0000000000
TOTAL ERROR = .0000000000

INDIVIDUAL SIGNAL PROBABILITY DISTRIBUTIONS

VAL.	1	2	3	4	5
1	1.0000000000	.9000000000	0.0000000000	0.0000000000	0.0000000000
2	0.0000000000	0.0000000000	1.0000000000	.7200000000	.5040000000
3	0.0000000000	.1000000000	0.0000000000	.2800000000	.4960000000

EXECUTION TIME = .14 SECONDS.

FIGURE 20 FINAL EVENT TABLE FOR EXAMPLE SYSTEM WITH ALL SIGNALS AS FINAL OUTPUTS

GO Symbols

In the prior sections of this chapter most of the symbols employed in the GO methodology have been introduced. In this section we summarize the conventions which have evolved.

The logical operator types without inputs (types 4 and 5) are represented in a GO model using triangles. All other operator types are represented with circles. Inside either the triangle or the circle representing a system element is written a hyphenated type-kind number identifying the logical type of the operator and the associated probabilities referenced by the kind number.

Model elements are connected by arrows showing the direction of flow. These arrows or lines are all numbered to identify the input and output signals to the operators. Half arrows are used to identify the second input for operators requiring two inputs where signal order is important.

Numbers are used as identifiers for each of the 15 logical operator types, as arbitrary kinds to reference probabilities, as signal numbers

specifying the random variables, and as the small set of defined values which the random numbers may take. In addition, not previously discussed, the operators are numbered by the sequence in which they are executed as listed in the G01 data to generate an operator number for each.

Events are specified by signal numbers subscripted with specific values. Joint events involving two or more random variables are identified by a string of such subscripted signal numbers. For example, the representation $8_1 12_3$ means the event that signal 8 takes value 1 and signal 12 takes value 3.

GO Terminology

To express ideas and concepts precisely a vocabulary of GO terminology has developed. A number of words have been given specific meanings. Experience has shown that these definitions permit consistent, unambiguous exposition of the procedure.

Thirty-three of the terms defined to enhance understanding about and use of the GO methodology are listed in Figure 21. A brief definition of each term is provided. A number of the terms have not been used in our exposition to this point. They await a full development in companion manuals for the more serious student and user of the GO methodology.

GO Supertypes

To conclude this brief exposition of the GO methodology we mention one additional feature, that of GO supertypes. A supertype is simply a defined collection of elemental GO operators which is treated as an entity and which may be called up one or more times in a GO model. Supertypes are used to reduce the complexity of a model (like a block diagram approach) or to eliminate the need to remodel identical configurations.

A supertype is defined using an identification number greater than or equal to 100. A flag of -1 or 0 is used to inform the computer whether this is a definition or a use of the supertype. The dummy input signal numbers in the supertype definition are allocated to the range 100-199. Dummy output signal numbers are allocated the range 200-299. Dummy kind numbers which can be used to permit different probabilities of operation for logically equivalent configurations must exceed 999.

To illustrate the concept consider the schematic of Figure 22.

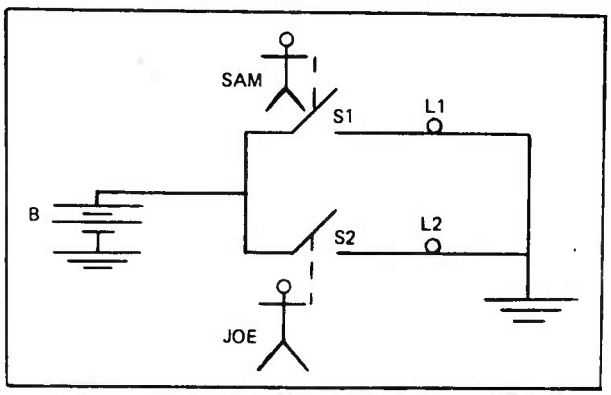


FIGURE 22 EXAMPLE SCHEMATIC

Active Signal: A signal which is included in the current probability distribution.

Branch Combination: The joining of several branches of an event tree when the branches differ only in the values of signals which are deleted at that stage. Mathematically a branch combination is equivalent to finding the joint marginal distribution of the active signals which results from summing out the deleted ones.

Deleted Signal: A signal which is deleted or dropped during the prncessing of an operator hy summing over its values. The resulting oistribution is the joint marginal distributinn of all of the active signals except the deleted one.

Distribution: The prnbability mass function of one or more signals. The adjective 'joint' may he used if more than one signal is involved and 'marginal' if the distribution has resulted from deleting one or more signal. Also called an "array".

Event Tree: The logical tree which shows the progression of the probabilistic analysis of a GO model. It is usually a conceptual device but, given time, patience, and a log to paper, can be drawn for any model.

Fault Set: The list of operators having critical states which lie on one branch of the selected event trees; i.e., a set of operators, which lead to the selected event when in their critical states.

Fault Finder Sequence: The program sequence GO1, GO2, GO3, FF1, FF2, FGO is sometimes appended to this sequence.

Final Distribution: The probability distribution which is printed by GO3. The signals (random variables) which are included in this distribution are designated by the user and normally are those whose values are of importance in describing the overall operation of the modeled system.

Final Event: An element of the final distribution. It is a statement of the particular values taken on by each of the final signals and has an associated probability.

Final Signal: A signal which is included in the final probability distribution produced by GO.

GO Sequence: The program seguence GO1, GO2, and GO3.

Improvability: The change in the probability of a final event prnduced by increasing the success probability of a variable kind to as high a value as possible.

Infinity: The largest possible value of a signal; defined as a program parameter by the user.

Input Signal: A signal whose values must be known by an operator for determining the values of its output signal(s).

Kind: Refers to the set of parameters (usually probabilities and signal values) which, together with its type designation, completely defined an operator. Each type has a particular set of required kind data. (Some types require none.)

Kind Number: An arbitrarily assigned number referencing the probabilities associated with an operator.

Never: = infinity.

Operator: The fundamental element of a GO model. It represents an algorithm for creating one or more new signals.

Operator Number: A seguential number assigned by the computer to reference the GO1 operator data.

Operator State: A mode of operation of an operator - for example, 'good', 'failed', or 'premature'. Each operator state produces a value for the operator's output signal(s). Each state has an associated probability, the probabilities summing to unity.

Output Signal: A signal created by an operator. The operator actually creates a new joint probability distribution which includes the output signals.

PMIN: A GO3 parameter specifying the term probability pruning value.

Program Parameter: A parameter which controls the operation of a GO program. All program parameters have default values which may be altered by the user of the appropriate parameter records.

Pruning: The deletion of a branch of an event tree at a point where the branch probability, up to that point, becomes very small (less than or equal to PMIN).

Selected Event: A user-defined logical combination of the final event. Used in Fault Finder programs.

Sensitivity: The rate of change of a final event probability with respect to the success probability of a variable kind.

Signal: A random variable created by an operator. The signal values and their probabilities are determined by the operator type, the associated kind data, and the values of the operator input signals, if any.

Signal Number: An arbitrarily but uniquely assigned number identifying random variables.

Signal Value: One of the possible integer values (between zero and "infinity") which can be assumed by a signal.

Supertype: A user-defined collection of operators connected together in some operationally meaningful manner and treated as a single entity.

Time Point: = signal value.

Type: An operator classification. Each of the 15 types is represented in the GO programs by an algorithm which defined its operational nature and combines and generates signals.

Value: One of a set of numbers (between 0 and INFIN) which the random variables take.

FIGURE 21 GO TERMINOLOGY

The system consists of a battery, two switches, and two lights. Assume that the battery is connected to the circuit at a certain time, a little later S1 is actuated, and even later S2 is actuated (we will designate these times by the numbers 1, 2, and 3 respectively). Assume that our interest is in the time at which at least one light comes on.

In creating a GO model for this system we observe that there are two redundant switch-light trains. Consequently a supertype modeling one train can be defined once and called up twice, once for each train. Let us call this supertype 100.

The GO1 data for our GO model of this schematic could be written as follows:

```
GO1 DATA FOR EXAMPLE SCHEMATIC
 $PARAM INFIN=4 $
100 -1 101 102 201 $ SUPERTYPE 100 DEFINITION
6 4 101 102 1 $ SWITCH
1 5 1 201 $ LIGHT
END*
5 1 1 $ BATTERY INSTALLATION
1 2 1 2 $ BATTERY
5 3 3 $ SAM
100 0 2 3 5 $ TOP SWITCH-LIGHT CHANNEL
5 4 4 $ JOE
100 0 2 4 6 $ BOTTOM SWITCH-LIGHT CHANNEL
2 0 2 5 6 7 $ OR GATE
0 7 $ FINAL SIGNAL
EOR
```

FIGURE 23 EXAMPLE SUPERTYPE DEFINITION AND USE

Note that the first instructions in GO1 define the switch-light supertype as supertype 100 with dummy inputs 101 and 102 and dummy output 201. Signal 101 represents the electrical input from the battery and signal 102 represents the mechanical switch action by either Sam or Joe. Signal 201 represents the light output. No dummy kind numbers permitting variable kinds were defined for this example.

After the supertype is defined the GO1 model data is entered. Note the two calls of supertype 100 and the final OR combination of the supertype outputs defining system operation.

In drawing GO models using supertypes we have adopted the convention of using rectangles to represent them. An important consideration is to insure that the proper signals replace the dummy parameters in the uses of the supertypes. That is, to insure accurate model representation, the supertype must be "hooked up" properly.

---

* A supertype definition is terminated by any word string beginning with E, e.g., END, EUREKA.

The GO model for this example schematic is shown as follows:

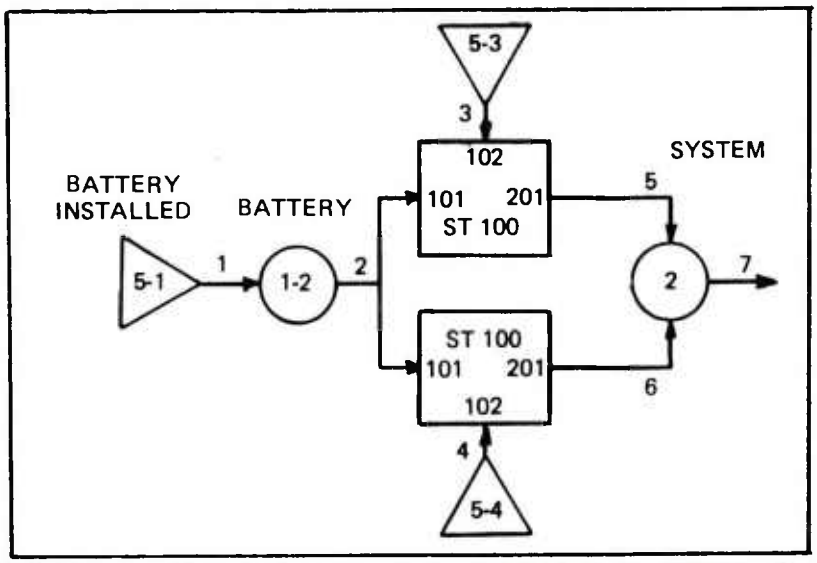


FIGURE 24 GO MODEL OF EXAMPLE SCHEMATIC USING SUPERTYPES

We will not pursue the supertype example further except to say that it is a very useful device when redundant or replicated models are created. The computer actually generates multiple unique sets of random variables for each supertype use. Consequently, the use of supertypes saves analyst labor but not computer processing time.

CHAPTER 4
EXAMPLES

This chapter portrays in detail the development of GO models of seven systems ranging from simple to relatively complex. The first example of a series system of two components is trivial and the system can be analyzed by hand in a few seconds. Nevertheless, the development of a GO model and the explanation and interpretation of the input data files and output printout for such an obvious system helps the reader grasp the mechanics of using the GO software. This example also addresses the objective of an assessment and the different types of data which can be utilized to generate system reliability, reliability with time, and availability estimates.

Example 2 treats a similar simple system composed of two components in parallel.

Example 3 explores the treatment of statistical dependencies in a GO model and conclusively demonstrates that the GO method properly accounts for such dependencies.

In each of these first three examples the probability trees which are generated by the GO software are explicitly shown. Most of the time the trees, while optionally available in table form, are not visible.

Example 4 is a somewhat more complex alarm system. It introduces the use of three values (time points) to represent system and component premature, success, and failure operational states. It also exemplifies the use of supertypes. (The alarm system example is addressed again in Chapter 5 when fault sets are treated.)

Example 5 portrays a comparison of the performance of a two-out-of-two pump train system with that of a two-out-of-three pump train system. The system is designed to continuously pump a specified amount of fluid at a given pressure. The advantage of introducing a third pump train is addressed.

The most complex example is that of a weapon fuzing system, example 6. The development of the GO model for this system takes only a couple hours. To analyze the system by hand or by other methods would require several man weeks or months of effort. Sixteen time points are used to characterize the safety and reliability of the weapon fuzing system. A supertype for a parallel pair of normally open switch contacts is used repeatedly to model the dual-channel cross-connected system.

The last example system analyzed, example 7, is that of a simple communication network. The probability of information flow from one node to another is calculated using a representative GO model. Possible two-way information flows on some links are modeled.

These examples provide a significant introduction to the range and application of the GO methodology and the specifics of generating data and interpreting the computer results.

Example 1 - Two Components In Series

The purpose of this example is to introduce the reader to the manner of developing a GO model, writing the GO input data, and interpreting the output data using a trivial example. Having introduced the reader to the GO process, the model is executed with three different types of data -- binomial attribute data, reliability with time data, and availability data to generate different system performance measures. The event tree for the model as developed by the GO software is shown. To conclude the example, alternate, but equivalent, ways of developing the system GO model are portrayed.

Consider the configuration of a system comprised of two components in series as depicted in Figure 25. Obviously both components must function for the system to function.

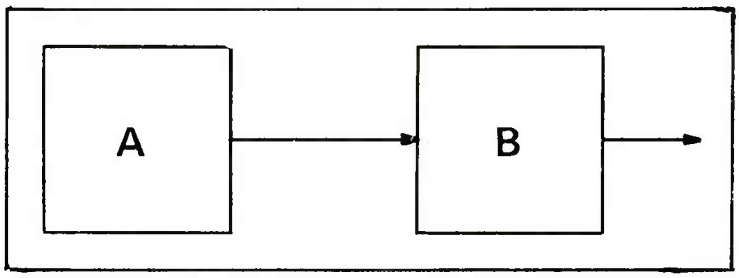


FIGURE 25
SERIES SYSTEM OF TWO COMPONENTS

Consequently, we can write the equation for the reliability of this system by inspection. It is a function of the reliability of each component. If r_A represents the reliability of component A, r_B that of component B, and r_S that of the system, then

$$r_S = r_A \bullet r_B \ . \qquad (1)$$

Similarly, we can write the equation for the availability of this system. If a_A represents the availability of component A, a_B that of component B, and a_s that of the system, then

$$a_s = a_A \bullet a_B \ . \qquad (2)$$

We will use these "obvious" equations later to compare the system reliability and availability with the results obtained from several GO models. We now develop a GO model of this system which we will subsequently execute with three different types of data. Our objective in each case will be to quantitatively determine the reliability or the availability of the system.

In Figure 26 we have a representation of this system using standardized GO operators and symbols.

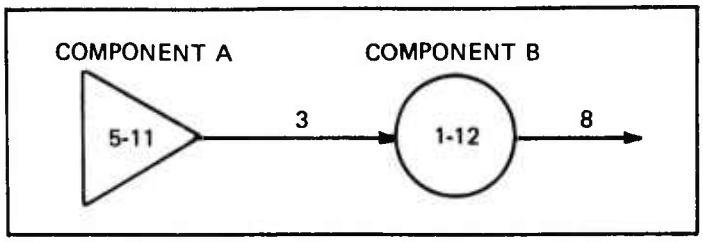


FIGURE 26 GO MODEL OF SERIES SYSTEM OF TWO COMPONENTS

In this model we have used a type 5 operator which requires no inputs and produces one output to represent component A. The output signal generated is called signal #3. It becomes the input to component B which is modeled as a type 1 operator which has one input signal, signal #3, and one output signal, signal #8. Signal #8 is the output signal of interest for this system model. (The numbering of the signals is completely arbitrary in the range 1-1500, but signal numbers must be unique.)

Initially we will permit the signals (random variables) to take values of 0 or 1. A value of 0 will mean successful -- reliable or available -- and a value of 1 will mean failure -- unreliable or unavailable. Consequently, the event 8_0, signal 8 takes value 0, represents system success. The event 8_1, signal 8 takes value 1, represents system failure.

The probabilities associated with the operational state of the two components, A and B, are referenced by the kind numbers 11 and 12 respectively. (The kind numbers selected are again completely arbitrary in the range 1-200.)

As a first case the reliabilities of the two components have been obtained by testing samples of components from the populations of components from which A and B have been randomly selected to fabricate the system. For component A, 33 like components were tested, 31 were reliable. For component B, 74 have been tested and all but one were found to be successful.

Consequently, the reliabilities for components A and B are the proportions:

$$r_A = \frac{31}{33} = 0.9394, \quad r_B = \frac{73}{74} = 0.9865.$$

To use the KSC GO software to calculate the reliability of this system, three data records, one for each of the three GO programs, GO1, GO2, and GO3, are prepared. The data record for GO1 communicates the logical structure of the system -- its configuration defining components and their interactions -- to the computer. The data record for GO2 communicates the point estimate probabilities associated with each model element. The data record for GO3 communicates information influencing the manner in which the probability tree for the system will be created.

Figure 27 records the three data files for this system. The records are separated with "EOR" which means End of Record.

```
GO1 DATA FOR SERIES SYSTEM OF TWO COMPONENTS
 $PARAM INFIN=1$
5 11 3 $ COMPONENT A
1 12 3 8 $ COMPONENT B
0 8 $ FINAL SIGNAL
EOR
GO2 DATA FOR SERIES SYSTEM OF TWO COMPONENTS
 $PARAM $
11 5 2 0 0.9394 1 0.0606 $ COMPONENT A
12 1 0.9865 0.0135 $ COMPONENT B
EOR
GO3 DATA FOR SERIES SYSTEM OF TWO COMPONENTS
 $PARAM PMIN=0.0 $
EOR
```

FIGURE 27 GO DATA FOR SERIES SYSTEM OF TWO COMPONENTS

In each record the first line entry records the name or title given the record and the second line entry calls up a Fortran namelist parameter called PARAM which allows the user to set selected parameter values. In GO1 the parameter INFIN was set to 1, permitting the signals to take the values 0 and 1. Consequently, in this model, 1 is the largest permissible value. In GO2 no parameters were changed so default values were used. In GO3 parameter PMIN was set to 0. Consequently, no terms will be discarded or pruned.

In the GO1 data record we next introduced the data line "5 11 3 $ COMPONENT A". This tells the computer that the first operator is a type 5 operator with kind 11, probabilities generating signal #3. The $ (dollar sign) terminates the entry and we can further identify the model element with descriptive information to the right of the terminator. Next, the information for component B is entered. Component B has been modeled as a type 1, kind 12, operator with input signal #3 and output signal #8. The last entry "0 8 $ FINAL SIGNAL" terminates the GO1 data input. The 0 is a flag indicating no additional operators are to be specified. The subsequent entries on the data line initialized with a 0 tell the computer the final signals to be included in the output data and the order in which they will occur in the output joint distribution. In this case only the single signal, #8, is specified.

The GO1 software recognizes the nature of the data which must be entered for each logical operator type. (See Figure 11, GO1 Operator Data, in Section 3.) Diagnostics are provided if the user fails to provide the requisite amount of data or if the data supplied is inconsistent.

The GO2 data record consists of a name line and a parameter line followed by two line entries recording the probability data for kinds 11 and 12. The line entry "11 5 2 0 0.9394 1 0.0606 $ COMPONENT A" records that kind 11 is used by a type 5 operator which generates a signal taking 2 values. These values are 0 which is taken with probability 0.9394 and 1 which is taken with probability 0.0606. These probabilities, specifying the probabilities with

which the possible values are taken, must sum to 1.0. The dollar signal ($) is again a terminator to the right of which descriptive information may be written.

The line entry for kind 12 reads "12 1 0.9865 0.0135 $ COMPONENT B". It specifies that kind 12 data is used by a type 1 operator. A type 1 operator requires two probabilities -- the first for the success mode and the second for the failure mode. These again must sum to 1.0.

The GO2 software recognizes the nature of the probability data which must be entered for each logical operator type. (See Figure 14, GO2 Kind Data, in Section 3.) Diagnostics are provided if the user fails to provide the requisite amount data or if the data supplied is inconsistent.

The GO3 data record for this model consists of only two data lines, the name, and the parameter, entries.

The output files from executing the GO1, GO2, and GO3 software using the input data of Figure 27 are shown in Figures 28, 29, and 30.

Figure 28 portrays the output file for GO1 for the Series System of Two Components defined by the data described above. The output records the date on, and time at, which this run was executed and notes the software version being used. It then records the name of the model as noted in the GO1 name line entry. Next the values of the various parameters are recorded. We set INFIN equal to 1. Default values were used for other parameters. (These are defined and explained in the KSC GO Reference Manual.)

The operator data furnished is recorded and each operator is numbered. The software then provides a cross-reference index documenting where signals are created and where they are used. For example, in this model signal #3 was generated by operator number 1 which is a type 5, kind 11 operator. Signal #3 is used in operator number 2 and, since it is not specified as a final output signal and is no longer needed as an input to any other operator, it is deleted from further consideration.

The next set of information on the GO1 printout, Figure 28, registers the number of active signals in the distribution after each operator has been executed. The entry 1(1) means that after operator 1 has executed, there is only one signal in the distribution. Reference to the GO model of Figure 26 or to the data for operator number 1 shows that this is signal #3. After the second operator has executed, there is again only one signal in the distribution. This time it is signal #8.

Some additional information about the model is then recorded, e.g., numbers of operators, signals, etc. Then the final signals which will appear in the output distribution are noted (in this case only signal #8 will appear).

```
GO1  (KSCGO, VAX VERSION 1.0)  RUN ON 20-OCT-82 AT 10:43:54
COPYRIGHT (C) 1982 by KAMAN SCIENCES CORPORATION

GO1 DATA FOR SERIES SYSTEM OF TWO COMPONENTS

INFIN =   1, VALUES =   2, BIAS =  750, OPS = 1, SIGNALS = 1,
                                                 ERRORS = 25
 OP    DATA
----   ------------------------------------------------------------
   1   5 11 3 $ COMPONENT A
   2   1 12 3 8 $ COMPONENT B
///3   0 8 $ FINAL SIGNAL

SIGNAL DATA

         SOURCE OPER.
SIGNAL   NUM TYPE KIND   USING OPERATORS (- IF DELETED AT)
------ ---- ---- ----   ----------------------------------------
    3     1    5    11     -2
    8     2    1    12

OPERATOR(NUMBER OF ACTIVE SIGNALS)
-----------------------------------
   1(  1)     2(  1)

NUMBER OF OPERATORS... =     2
NUMBER OF SIGNALS..... =     2
MAXIMUM NUMBER ACTIVE. =     1
MAX SIGNAL LIST SIZE.. =     1
NUMBER OF SIGNALS/WORD =    32
MAXIMUM WORDS/TERM.... =     1

FINAL SIGNALS =      8

 CPU TIME = 0:00:00.39
 DIRECT I/O COUNT = 5
 ELAPSED TIME = 00:00:01.37
```

FIGURE 28

GO1 DATA FOR SERIES SYSTEM OF TWO COMPONENTS

```
GO2  (KSCGO, VAX VERSION 1.0)  RUN ON 20-OCT-82 AT 10:43:56
COPYRIGHT (C) 1982 by KAMAN SCIENCES CORPORATION

GO2 DATA FOR SERIES SYSTEM OF TWO COMPONENTS

PERF = 0, MXSIZE = 2048

OPERATOR FILE --- GO1 DATA FOR SERIES SYSTEM OF TWO COMPONENTS

RECORD KIND DATA
------ ------------------------------------------------------------
     1  11 5 2 0 0.9394 1 0.0606 $ COMPONENT A
     2  12 1 0.9865 0.0135 $ COMPONENT B
     3  END
------------------------------------------------------------------

USE SUMMARY TABLE, ENTRY = KIND/TYPE(FREQUENCY)
   (FREQUENCY IS NEGATIVE FOR PERFECT KINDS.)

 11/ 5(     1)   12/ 1(     1)

NUMBER OF KINDS INPUT-----    2
NUMBER USED - NONPERFECT--    2
NUMBER USED - PERFECT-----    0

   1 FILE RECORDS WRITTEN FOR      2 OPERATORS.

 CPU TIME = 0:00:00.40
 DIRECT I/O COUNT = 13
 ELAPSED TIME = 00:00:01.35
```

FIGURE 29
GO2 DATA FOR SERIES SYSTEM OF TWO COMPONENTS

```
G03  (KSCGO, VAX VERSION 1.0)  RUN ON 20-OCT-82 AT 10:43:58
COPYRIGHT (C) 1982 by KAMAN SCIENCES CORPORATION

G03 DATA FOR SERIES SYSTEM OF TWO COMPONENTS

OPERATOR FILE --- G01 DATA FOR SERIES SYSTEM OF TWO COMPONENTS
KIND FILE ------- G02 DATA FOR SERIES SYSTEM OF TWO COMPONENTS

RUN NUMBER  1
     PMIN =0.0000E+00
     NEW = 0, INTER = 0, SAVE = 0, MXDIST =  3000
     FIRST = 10000, LAST = 10000, TRACE = 2.00000

-----------------------------------------
FINAL EVENT TABLE (INFINITY =   1)

                SIGNALS AND THEIR VALUES
                ------------------------
PROBABILITY       8
--------------  ----
 0.0732819000     1
 0.9267181000     0

-----------------------------------------

TOTAL PROBABILITY =  1.0000000000
TOTAL ERROR =        0.0000000000


INDIVIDUAL SIGNAL PROBABILITY DISTRIBUTIONS

VAL.         8
-----  -------------
  0    0.9267181000
  1    0.0732819000


 CPU TIME = 0:00:00.36
 DIRECT I/O COUNT = 7
 ELAPSED TIME = 00:00:00.69
```

FIGURE 30
GO3 DATA FOR SERIES SYSTEM OF TWO COMPONENTS

Finally the CPU time (central processor units), direct I/O count (number of input/output data records of size 512 bytes manipulated), and the elapsed clock time required to execute program GO1 for this model are recorded.

The GO2 output data (Figure 29) is similar to that of GO1. The date and time of execution, the software version being used, the name of the model, and the parameter values are all recorded. The kind data is then listed and each entry numbered. A use summary table for each kind/type is provided recording how many operators referenced each set of kind probabilities. Some additional details about the data are provided, then the CPU time, I/O count, and elapsed time for program GO2 execution are recorded.

The GO3 output data (Figure 30) contains the results from executing this series system of two components. After the copyright notice indicating the software version being used, the title of this system as entered on the first line of GO3 data is printed. This is followed by an identification of the operator (GO1) and kind (GO2) files used as input to GO3. Next the values of the parameters used in GO3 are given. With the exception of PMIN which was set equal to 0.0, all other parameters have their default values. (The meaning and purpose of these additional parameters are explained in the GO Reference Manual.)

In Figure 30, the Final Event Table lists the two possible events: 8_0, the system success event, and 8_1, the system failure event. The probability of success (system reliability) is 0.9269181 and that of failure, 0.0732819.

In this execution no terms were discarded so there is no computational error. The line entry "Total Probability = 1.0" is the sum of the probabilities of occurrence of all events. Subtracting that sum from unity gives zero computational error due to pruning for this execution.

For many system models more than one final output signal will be requested. Consequently, the Final Event Table will contain the joint probability distribution of all such final signals. Subsequently the computer generates the Individual Signal Probability Distributions for each of these signals. In this case the probability distribution for signal #8 is repeated in slightly different format.

The GO3 output also lists the CPU time for execution, the direct I/O count, and the elapsed clock time.

As noted above the reliability for this series system of two components was calculated by GO3 to be 0.9267181. This value can be obtained directly from equation 1, page 34 where

$$r_s = r_A \bullet r_B,$$

$$r_s = 0.9394 \bullet 0.9865 = 0.9267181.$$

This substantiates the result generated from the GO software. Of course, for such a trivial problem the GO methodology is not required, and it is only used

here for illustrative purposes. The benefits from employing GO become significant when one cannot write the system success equation by inspection.

The event tree which is developed by the GO model for this system is shown in Figure 31. After the operator representing component A is executed in GO3 two branches exist, the upper branch with event 3_0 and the lower branch with event 3_1. Then, the second operator representing component B is introduced. Four branches are generated. One represents the success event, 8_0. The other three branches lead to system failure, event 8_1. They are combined and their probabilities summed to generate the probability of system failure. (This can be done because the different branches of the tree are mutually exclusive.)

Without changing the logical configuration of this system, we will calculate another reliability performance measure based on different data. In the execution of this model as treated above, we calculated system reliability from component reliabilities which were based upon the proportion of successes obtained in n trials. Time was not a factor in these estimates, and consequently, the system reliability estimate is, without further information, time invariant.

In contrast, we now postulate that the components are degrading with time, the degradation being specified by failure rates of $6.27 \times 10^{-}$ failures/hour and $3.42 \times 10^{-}$ failures/hour for components A and B respectively. By specifying the failure rate we mean that the components exhibit an exponential time-to-failure distribution. In this instance time is very much a parameter. Now the proper question is, "what is the system reliability after so many hours of service assuming no repair?"

To calculate the system reliability at, say, 5000 hours, we first calculate the component reliabilities at that time. Since,

$$r_i = e^{-\rho_i t}$$

where r_i is the reliability, and ρ_i the failure rate of the i^{th} component. The reliability is calculated at time t. Hence,

$$r_A = e^{-6.27 \times 10^{-6} \bullet 5 \times 10^{3}} = 0.9691$$

$$r_B = e^{-3.42 \times 10^{-6} \bullet 5 \times 10^{3}} = 0.9830$$

When this data is entered in GO2 and the system model executed, the system reliability at 5000 hours is calculated to be 0.9526253. Obviously the system reliability can be obtained for any point in time. A number of such estimates were calculated to generate the curve of Figure 32. (This was done using the repetitive option in GO3 - see GO Reference Manual for details.)

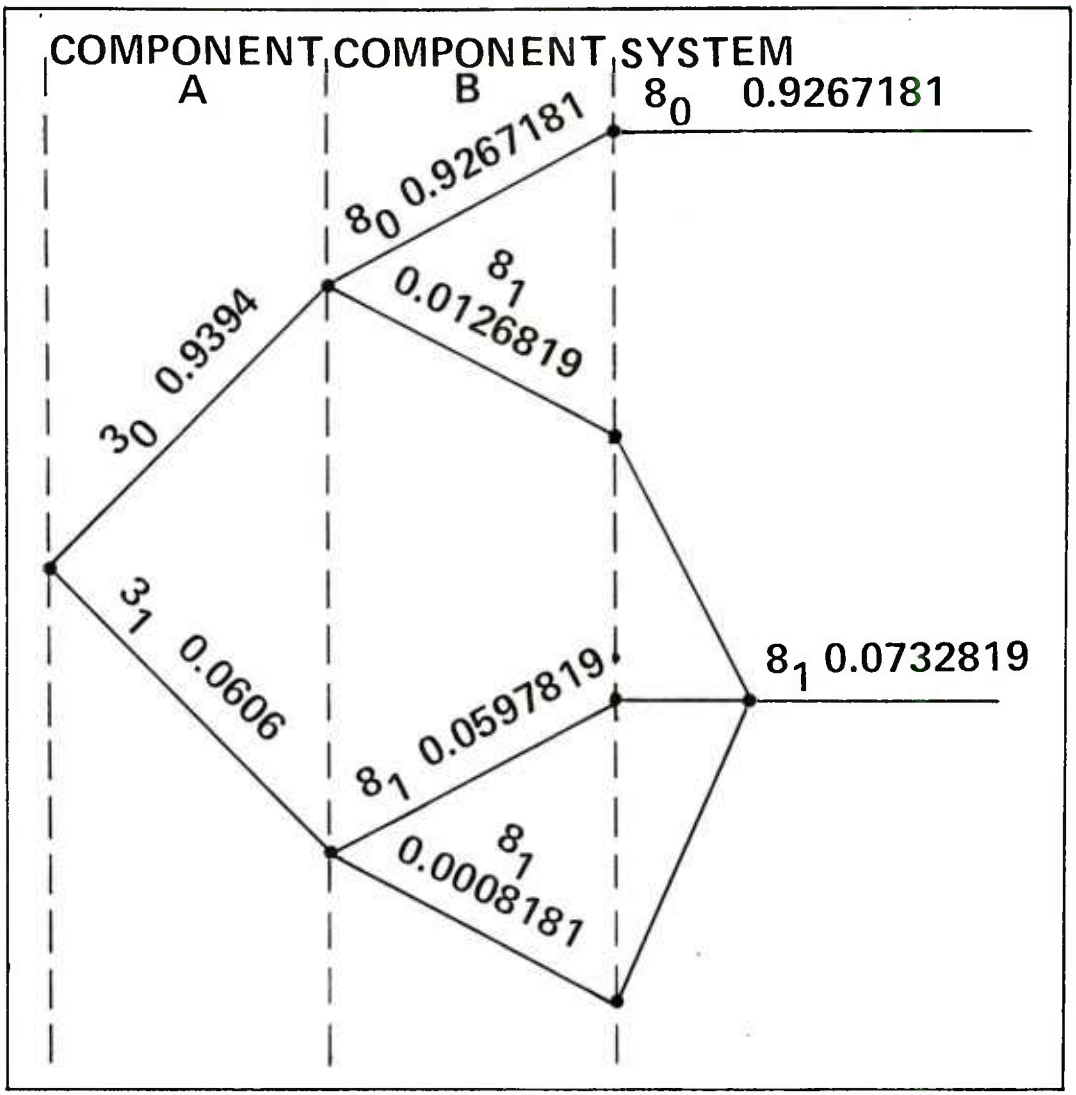


FIGURE 31
EVENT TREE FOR SERIES SYSTEM OF TWO COMPONENTS

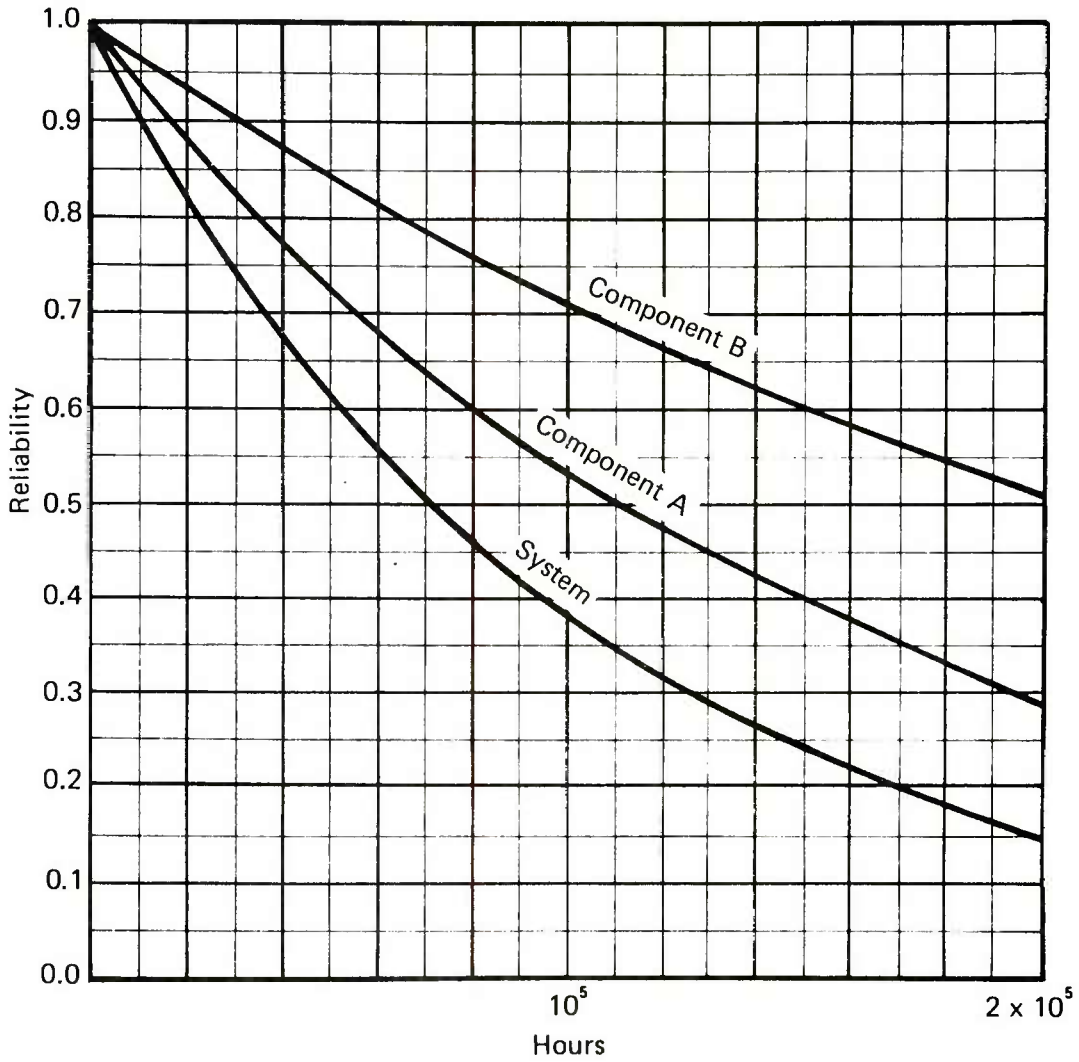


FIGURE 32 RELIABILITY WITH TIME, NO REPAIR, SERIES SYSTEM OF TWO COMPONENTS

As a final variation of the data which might be used to calculate a performance measure for the series system of two components, let us now postulate that when a component fails it is detected immediately and repaired. We now seek the proportion of the time the components and the system are functional, e.g., their availability.

To generate the availability point estimates for the components we need some information about the length of time repairs take. Knowing that the average repair times, or, more commonly, the mean-times-to-repair (MTTR), for component A and B are four and 16 hours respectively, for example, permits us to calculate the availability of each component. This is done using the standard formula

$$A = \frac{MTTF}{MTTF + MTTR} .$$

The MTTF can be obtained as the reciprocal of the failure rate. Using the failure rates specified above, the MTTF for components A and B are 159489.6 and 292397.7 hours respectively. The availabilities of components A and B are then:

$$a_A = \frac{159489.6}{159489.6+4} = 0.999975$$

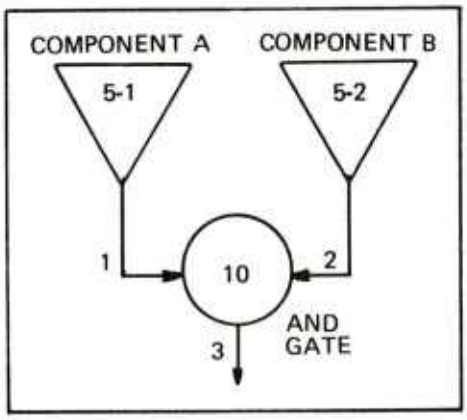
$$a_B = \frac{292397.7}{292397.7+16} = 0.999945$$

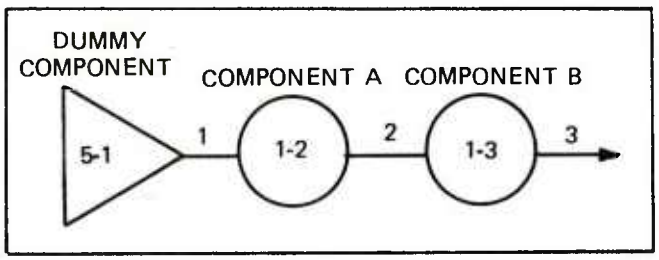
Using these probabilities as the component data for GO2, the system availability, a_S, can be calculated. It is, 0.999920. (One can use equation 4.2 to calculate this result for comparison.) The effect of repairs upon system performance is seen to be pronounced. The probability that the system is functional at any random point in time (its availability) is 0.999920, whereas, without repair, the reliability degrades quite rapidly with time as previously noted in Figure 32.

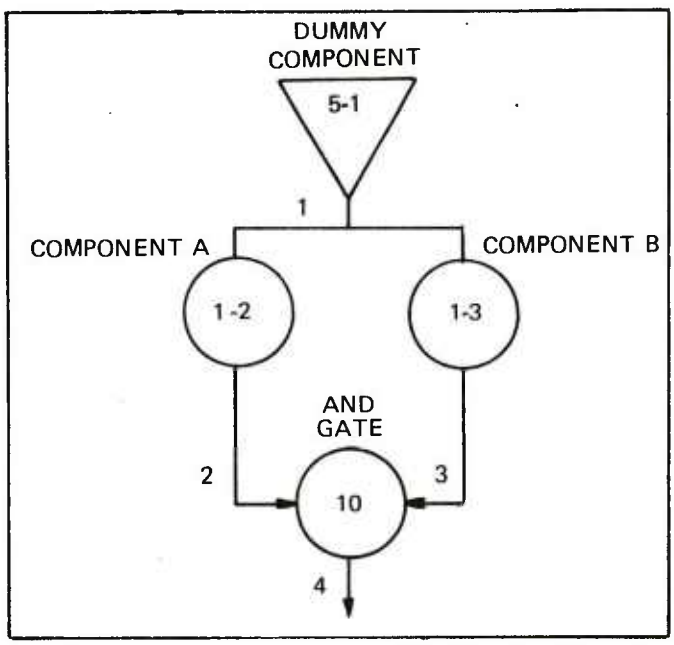
We now observe that the GO model of Figure 26 for this series system of two components is not unique. Shown in Figure 33 are four additional ways the model could have been constructed to generate identical results.

The preparation of the data for these GO models is left as an exercise for the reader. Refer to Figures 11 and 14 for the GO1 and GO2 data formats for different operator types.

In this example we have demonstrated how a GO model for this simple two-component system is developed. We executed the model with three different types of data - bionomial attribute data, reliability with time data, and availability data. The model input and output data were shown and explained in detail. Finally we demonstrated other ways the system could have been modeled.







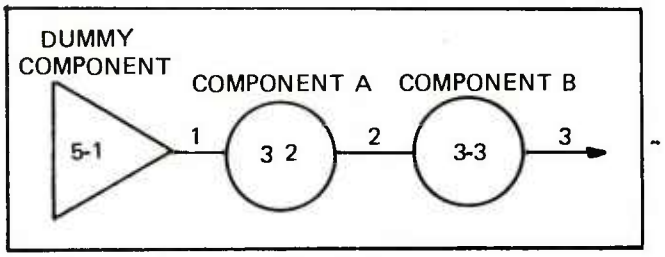


FIGURE 33
EQUIPMENT GO MODELS FOR SERIES SYSTEM
OF TWO COMPONENTS

Example 2 - Parallel System of Two Components

This example is an extension of Example 1. Its purpose is to portray another simple example so the reader gains familiarity with the procedure on models he can easily comprehend and check. The same components and data will be used, but the system configuration is now a simple parallel system in contrast with the series system of Example 1. Figure 34 depicts the system. If either component is successful, the system is successful.

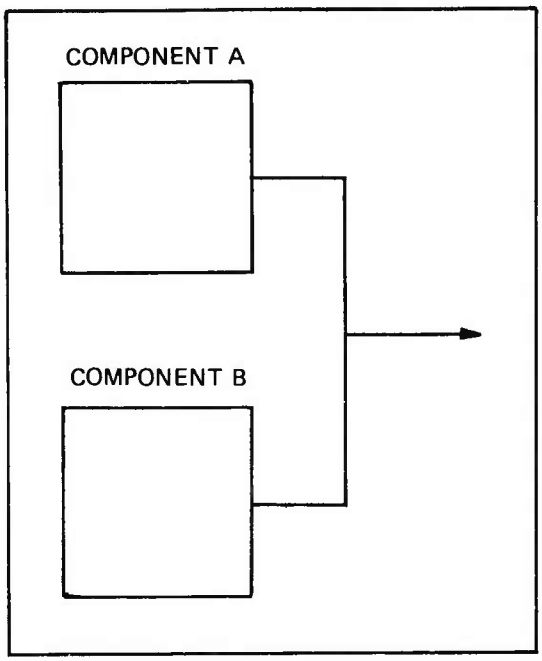


FIGURE 34 PARALLEL SYSTEM OF TWO COMPONENTS

To calculate system reliability the GO model of Figure 35 is constructed.

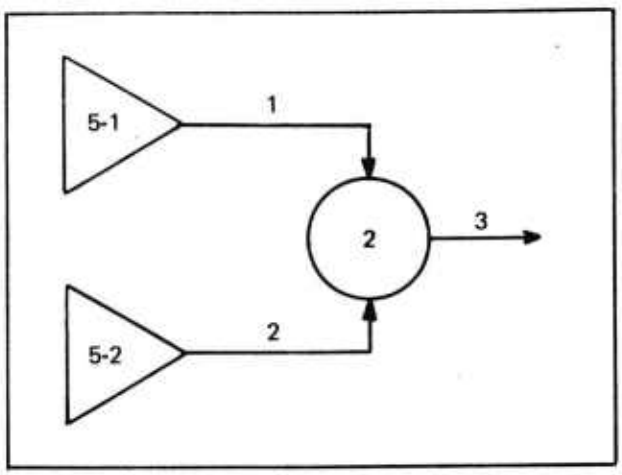


FIGURE 35 GO MODEL OF PARALLEL SYSTEM OF TWO COMPONENTS

The GO data to execute this model is listed in Figure 36.

```
GO1 DATA FOR PARALLEL SYSTEM OF TWO COMPONENTS
 $PARAM INFIN=1$
5 1 1 $ COMPONENT A
5 2 2 $ COMPONENT B
2 0 2 1 2 3 $ OR GATE
0 3 $ FINAL SIGNAL
EOR
GO2 DATA FOR PARALLEL SYSTEM OF TWO COMPONENTS
 $PARAM $
1 5 2 0 0.9394 1 0.0606 $ COMPONENT A
2 5 2 0 0.9865 1 0.0135 $ COMPONENT B
EOR
GO3 DATA FOR PARALLEL SYSTEM OF TWO COMPONENTS
 $PARAM PMIN=0.0 $
EOR
```

FIGURE 36 GO DATA FOR PARALLEL SYSTEM OF TWO COMPONENTS

The results from executing this model by the GO1, GO2, and GO3 software are recorded in Figures 37, 38, and 39 respectively. Of most interest is the reliability of the parallel system. From the Final Event Table of Figure 39, the calculated system reliability is 0.9991819. One can double check this result from the formula,

$$r_s = r_a + r_b - r_a \bullet r_b = 0.9394 + 0.9865 - 0.9267181 = 0.9991819.$$

Example 3 - Treatment of Dependencies

The purpose of this example is to explore the treatment of statistical dependencies in a GO model. Such dependencies are common factors in most systems. Often two or more components rely upon common power sources, common human actions, common environments, common water supplies, etc. The proper accommodation of such dependencies is crucial to the assessment of most real systems. This example demonstrates that GO models easily accommodate dependencies and that the GO software properly calculates the probabilities of occurrence of dependent events.

Consider the system whose GO model is shown in Figure 40. Two pumps each have a common water source and a common power source. The model documents that the output from both pumps is required for system success by requiring both inputs to an "AND" gate to generate system success.

```
GO1  (KSCGO, VAX VERSION 1.0)  RUN ON 19-OCT-82 AT 15:33:26
COPYRIGHT (C) 1982 by KAMAN SCIENCES CORPORATION

GO1 DATA FOR PARALLEL SYSTEM OF TWO COMPONENTS

INFIN =   1, VALUES =   2, BIAS =  750, OPS = 1, SIGNALS = 1,
                                                  ERRORS = 25
 OP    DATA
----   ------------------------------------------------------------
   1   5 1 1 $ COMPONENT A
   2   5 2 2 $ COMPONENT B
   3   2 0 2 1 2 3 $ OR GATE
///4   0 3 $ FINAL SIGNAL

SIGNAL DATA

        SOURCE OPER.
SIGNAL  NUM TYPE KIND  USING OPERATORS (- IF DELETED AT)
------ ---- ---- ----  ------------------------------------
    1     1    5    1    -3
    2     2    5    2    -3
    3     3    2    0

OPERATOR(NUMBER OF ACTIVE SIGNALS)
----------------------------------
   1(  1)    2(  2)    3(  1)

NUMBER OF OPERATORS... =    3
NUMBER OF SIGNALS.... =    3
MAXIMUM NUMBER ACTIVE. =    2
MAX SIGNAL LIST SIZE.. =    2
NUMBER OF SIGNALS/WORD =   32
MAXIMUM WORDS/TERM... =    1

FINAL SIGNALS =     3

 CPU TIME = 0:00:00.43
 DIRECT I/O COUNT = 5
 ELAPSED TIME = 00:00:01.30
```

FIGURE 37
GO1 DATA FOR PARALLEL SYSTEM OF TWO COMPONENTS

```
GO2  (KSCGO, VAX VERSION 1.0)  RUN ON 19-OCT-82 AT 15:33:28
COPYRIGHT (C) 1982 by KAMAN SCIENCES CORPORATION

GO2 DATA FOR PARALLEL SYSTEM OF TWO COMPONENTS

PERF = 0, MXSIZE = 2048

OPERATOR FILE --- GO1 DATA FOR PARALLEL SYSTEM OF TWO COMPONENTS

RECORD KIND DATA
------ ------------------------------------------------------------
     1   1 5 2 0 0.9394 1 0.0606 $ COMPONENT A
     2   2 5 2 0 0.9865 1 0.0135 $ COMPONENT B
------------------------------------------------------------------

USE SUMMARY TABLE, ENTRY = KIND/TYPE(FREQUENCY)
   (FREQUENCY IS NEGATIVE FOR PERFECT KINDS.)

  1/ 5(     1)     2/ 5(     1)

NUMBER OF KINDS INPUT-----    2
NUMBER USED - NONPERFECT--    2
NUMBER USED - PERFECT-----    0

   1 FILE RECORDS WRITTEN FOR      3 OPERATORS.

 CPU TIME = 0:00:00.35
 DIRECT I/O COUNT = 13
 ELAPSED TIME = 00:00:01.25
```

FIGURE 38 GO2 DATA FOR PARALLEL SYSTEM OF TWO COMPONENTS

```
GO3  (KSCGO, VAX VERSION 1.0)  RUN ON 19-OCT-82 AT 15:33:30
COPYRIGHT (C) 1982 by KAMAN SCIENCES CORPORATION

GO3 DATA FOR PARALLEL SYSTEM OF TWO COMPONENTS

OPERATOR FILE --- GO1 DATA FOR PARALLEL SYSTEM OF TWO COMPONENTS
KIND FILE ------- GO2 DATA FOR PARALLEL SYSTEM OF TWO COMPONENTS

RUN NUMBER  1
     PMIN =0.0000E+00
     NEW = 0, INTER = 0, SAVE = 0, MXDIST =  3000
     FIRST = 10000, LAST = 10000, TRACE = 2.00000

-----------------------------------------
FINAL EVENT TABLE (INFINITY =   1)

               SIGNALS AND THEIR VALUES
               ------------------------

PROBABILITY       3
-------------   ----
 0.0008181000     1
 0.9991819000     0

-----------------------------------------

TOTAL PROBABILITY =   1.0000000000
TOTAL ERROR =         0.0000000000


INDIVIDUAL SIGNAL PROBABILITY DISTRIBUTIONS

VAL.          3
----   ------------
  0    0.9991819000
  1    0.0008181000


 CPU TIME = 0:00:00.36
 DIRECT I/O COUNT = 7
 ELAPSED TIME = 00:00:00.79
```

FIGURE 39 GO3 DATA FOR PARALLEL SYSTEM OF TWO COMPONENTS

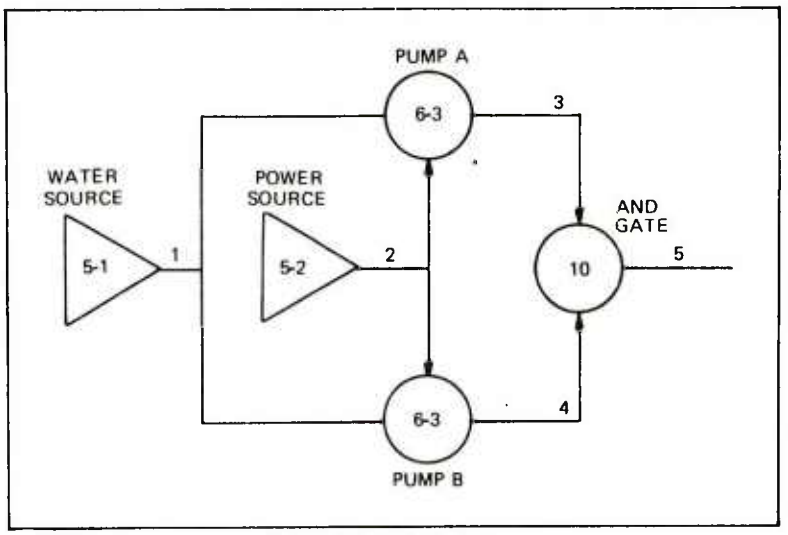


FIGURE 40
GO MODEL DEPENDENCIES BOTH PUMP TRAINS REQUIRED

It is obvious from the model that both pumps depend upon the common water and power sources. Consequently signals 3 and 4 are not independent. To show that the GO3 type 10 algorithm for an "AND" gate properly accommodates the dependencies of any input signals we will write the equations for system success and compare the probabilities of occurrences calculated by equations to the output from the GO software.

Let S_i be the event that signal i is successful. S_1 then represents the availability of water and S_2 the availability of power. Let P_A be the event that pump A is successful and likewise for P_B. The derived events S_3, S_4, and S_5 can now be expressed as a function of the elemental events. Set theory notation for the intersection of events is represented thus, $\cap$.

$$S_3 = S_1 \cap S_2 \cap P_A$$
$$S_4 = S_1 \cap S_2 \cap P_B$$
$$S_5 = S_1 \cap S_2 \cap P_A \cap P_B$$

If s_i now represents the probability of event S_i and p_A and p_B represent the probabilities of pumps A and B working properly, then the probabililties for events S_3, S_4, and S_5 can be expressed as follows:

$$P(S_3) = s_1 \bullet s_2 \bullet p_A,$$
$$P(S_4) = s_1 \bullet s_2 \bullet p_B,$$
$$P(S_5) = s_1 \bullet s_2 \bullet p_A \bullet p_B.$$

These equations demonstrate that the probability of occurrence of event S_5 is not the algebraic product of the probabilities of events S_3 and S_4. Because of their common dependencies upon the same power and water sources, the Boolean, rather than the algebraic, product of their probabilities of occurrence provides the correct probabilities of occurrence for event S_5.

Pursuing this example let the following quantitative estimates for the probabilities of occurrence of the elemental events previously defined be:

$$s_1 = 0.95$$
$$s_2 = 0.99$$
$$p_A = p_B = 0.9$$

Consequently the probabilities of S_3, S_4, and S_5 (to four places) are:

$s_3 = 0.8465,$
$s_4 = 0.8465,$
$s_5 = 0.7618.$

If the dependencies upon the common power and water sources had been ignored, as is often incorrectly postulated, and events S_3 and S_4 had been treated as if independent, the probability of event S_5 occurring would be computed to be $(0.8465)^2 = 0.7166$. The difference between this value and the correct value of 0.7618 is 0.0452, a nontrivial discrepancy.

This model was executed using the GO software. The permissable values the random variables may take are 0(success) and 1(failure). GO1, GO2, and GO3 data were developed from the information above. The results are shown in Figures 41, 42, and 43. Notice that the probability of event 5_0, which represents system success in Figure 43, is 0.7618, the correct answer. Consequently the GO software properly accounts for signal dependencies.

The event tree generated by the GO software for this model is shown in Figure 44. The tree depicts how the GO model generates the events and their probabilities of occurrence.

As a variation, redefine system success to be proper function of either pump. In this case, the type 10 in Figure 40 is replaced with a type 2 "OR" gate.

The event equation for system success is now:

$$S_5 = S_1 \cap S_2 \cap (P_A \cup P_B).$$

The probability of occurrence then becomes:

$$P(S_5) = s_1 \bullet s_2 \ (p_A + p_B - p_A \, p_B).$$

Using the same data as before,

$$P(S_5) = 0.95 \bullet 0.99 \ (0.9 + 0.9 - 0.81) = 0.9311.$$

If S_3 and S_4 were incorrectly considered to be independent, then the probability of system success would be 0.9764 calculated as follows:

$$P(S_5) = P(S_3 \cup S_4) = s_3 + s_4 - s_3 s_4 = 0.8465 + 0.8465 - 0.8465^2 = 0.9764.$$

Again the difference is not inconsequential, and the treatment of dependencies is seen to be important in developing system event probabilities.

Figures 45, 46, and 47 record the GO1, GO2, and GO3 output for system success with the new success criteria - either pump successful. Note that

```
GO1  (KSCGO, VAX VERSION 1.0)  RUN ON 20-OCT-82 AT 15:35:43
COPYRIGHT (C) 1982 by KAMAN SCIENCES CORPORATION

GO1 DATA FOR GO MODEL DEPENDENCIES - BOTH PUMPS

INFIN =   1, VALUES =   2, BIAS =  750, OPS = 1, SIGNALS = 1,
                                                 ERRORS = 25

 OP    DATA
----   ------------------------------------------------------------
   1   5 1 1 $ WATER SOURCE
   2   5 2 2 $ POWER SOURCE
   3   6 3 1 2 3 $ PUMP A
   4   6 3 1 2 4 $ PUMP B
   5   10 0 2 3 4 5 $ AND GATE
///6   0 5 $ FINAL SIGNAL

SIGNAL DATA

          SOURCE OPER.
SIGNAL    NUM TYPE KIND  USING OPERATORS (- IF DELETED AT)
------ ---- ---- ----  ----------------------------------------
     1      1    5      1      3     -4
     2      2    5      2      3     -4
     3      3    6      3     -5
     4      4    6      3     -5
     5      5   10      0

OPERATOR(NUMBER OF ACTIVE SIGNALS)
-----------------------------------
   1(  1)     2(  2)     3(  3)     4(  2)     5(  1)

NUMBER OF OPERATORS... =     5
NUMBER OF SIGNALS..... =     5
MAXIMUM NUMBER ACTIVE. =     3
MAX SIGNAL LIST SIZE.. =     3
NUMBER OF SIGNALS/WORD =    32
MAXIMUM WORDS/TERM.... =     1

FINAL SIGNALS =      5

 CPU TIME = 0:00:00.56
 DIRECT I/O COUNT = 6
 ELAPSED TIME = 00:00:02.29
```

FIGURE 41 GO1 DATA FOR GO MODEL DEPENDENCIES - BOTH PUMP TRAINS REQUIRED

```
GO2  (KSCGO, VAX VERSION 1.0)  RUN ON 20-OCT-82 AT 15:35:47
COPYRIGHT (C) 1982 by KAMAN SCIENCES CORPORATION

GO2 DATA FOR GO MODEL DEPENDENCIES - BOTH PUMPS

PERF = 0, MXSIZE = 2048

OPERATOR FILE --- GO1 DATA FOR GO MODEL DEPENDENCIES - BOTH PUMPS

RECORD KIND DATA
------ ------------------------------------------------------------
     1  1 5 2 0 0.95 1 0.05 $ WATER SOURCE
     2  2 5 2 0 0.99 1 0.01 $ POWER SOURCE
     3  3 6 0.9 0.1 0.0 $ PUMP
------------------------------------------------------------------

USE SUMMARY TABLE. ENTRY = KIND/TYPE(FREQUENCY)
   (FREQUENCY IS NEGATIVE FOR PERFECT KINDS.)

  1/ 5(     1)     2/ 5(     1)     3/ 6(     2)

NUMBER OF KINDS INPUT-----    3
NUMBER USED - NONPERFECT--    3
NUMBER USED - PERFECT-----    0

   1 FILE RECORDS WRITTEN FOR      5 OPERATORS.

 CPU TIME = 0:00:00.47
 DIRECT I/O COUNT = 13
 ELAPSED TIME = 00:00:02.46
```

FIGURE 42 GO2 DATA FOR GO MODEL DEPENDENCIES - BOTH PUMP TRAINS REQUIRED

```
GO3  (KSCGO, VAX VERSION 1.0)  RUN ON 20-OCT-82 AT 15:35:51
COPYRIGHT (C) 1982 by KAMAN SCIENCES CORPORATION

GO3 DATA FOR GO MODEL DEPENDENCIES- BOTH PUMPS

OPERATOR FILE --- GO1 DATA FOR GO MODEL DEPENDENCIES - BOTH PUMPS
KIND FILE ------- GO2 DATA FOR GO MODEL DEPENDENCIES - BOTH PUMPS

RUN NUMBER  1
     PMIN =0.0000E+00
     NEW = 0, INTER = 0, SAVE = 0, MXDIST =  3000
     FIRST = 10000, LAST = 10000, TRACE = 2.00000

-----------------------------------------
FINAL EVENT TABLE (INFINITY =    1)

                SIGNALS AND THEIR VALUES
                ------------------------
PROBABILITY        5
-------------   ----
 0.2381950000      1
 0.7618050000      0

-----------------------------------------

TOTAL PROBABILITY =   1.0000000000
TOTAL ERROR =         0.0000000000


INDIVIDUAL SIGNAL PROBABILITY DISTRIBUTIONS

VAL.          5
----   ------------
  0    0.7618050000
  1    0.2381950000


 CPU TIME = 0:00:00.36
 DIRECT I/O COUNT = 6
 ELAPSED TIME = 00:00:00.88
```

FIGURE 43 GO3 DATA FOR GO MODEL DEPENDENCIES - BOTH PUMP TRAINS REQUIRED

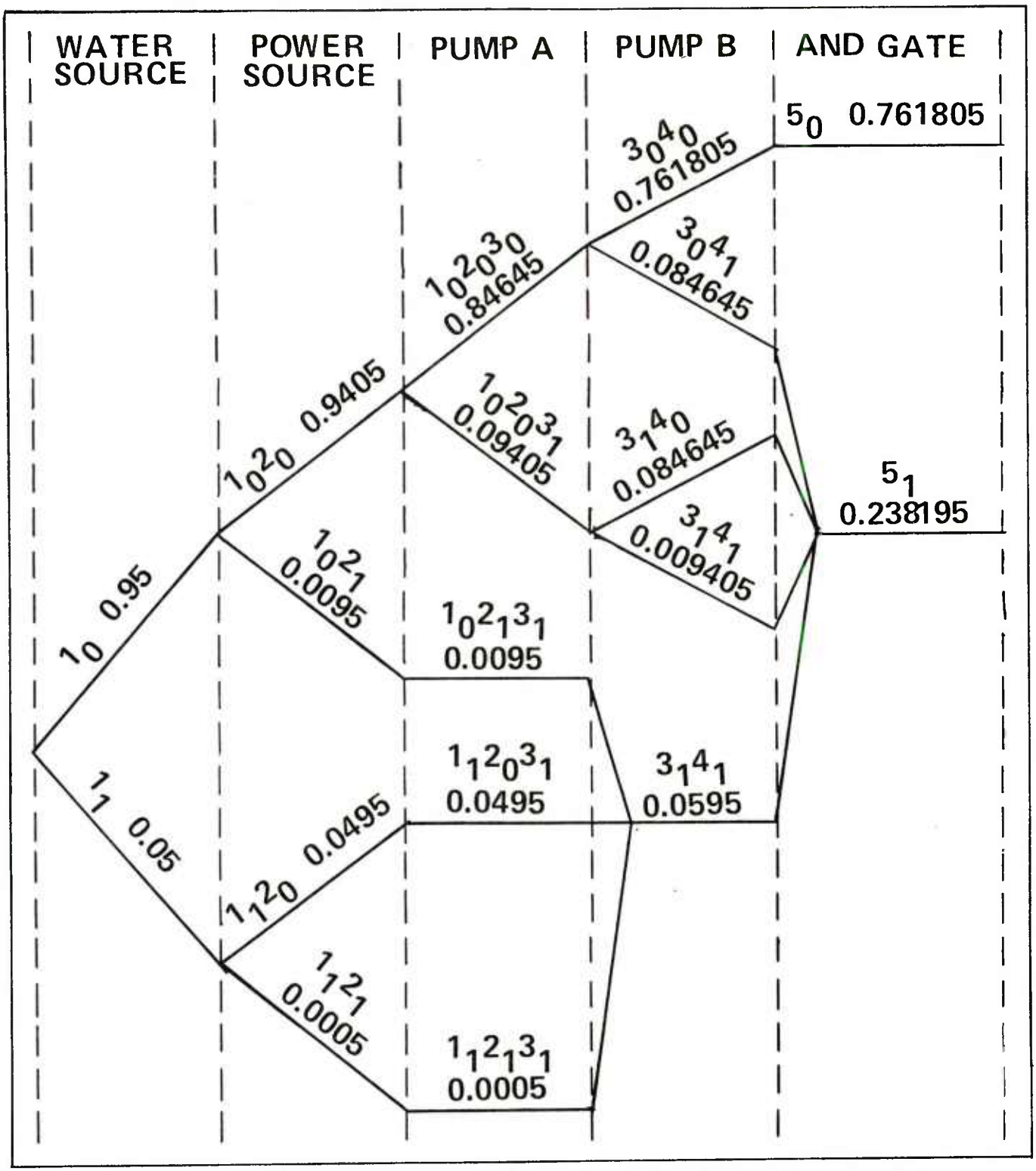


FIGURE 44 EVENT TREE FOR GO MODEL DEPENDENCIES - BOTH PUMP TRAINS REQUIRED

```
GO1  (KSCGO, VAX VERSION 1.0)  RUN ON 20-OCT-82 AT 15:42:44
COPYRIGHT (C) 1982 by KAMAN SCIENCES CORPORATION

GO1 DATA FOR GO MODEL DEPENDENCIES - EITHER PUMP

INFIN =   1, VALUES =   2, BIAS =  750, OPS = 1, SIGNALS = 1,
                                                 ERRORS = 25
 OP    DATA
----   ------------------------------------------------------
   1   5 1 1 $ WATER SOURCE
   2   5 2 2 $ POWER SOURCE
   3   6 3 1 2 3 $ PUMP A
   4   6 3 1 2 4 $ PUMP B
   5   2 0 2 3 4 5 $ OR GATE
///6   0 5 $ FINAL SIGNAL

SIGNAL DATA

         SOURCE OPER.
SIGNAL   NUM TYPE KIND   USING OPERATORS (- IF DELETED AT)
------- ---- ---- ----   ------------------------------------
    1      1    5      1      3     -4
    2      2    5      2      3     -4
    3      3    6      3     -5
    4      4    6      3     -5
    5      5    2      0

OPERATOR(NUMBER OF ACTIVE SIGNALS)
----------------------------------
   1(  1)     2(  2)     3(  3)     4(  2)     5(  1)

NUMBER OF OPERATORS... =     5
NUMBER OF SIGNALS..... =     5
MAXIMUM NUMBER ACTIVE. =     3
MAX SIGNAL LIST SIZE.. =     3
NUMBER OF SIGNALS/WORD =    32
MAXIMUM WORDS/TERM.... =     1

FINAL SIGNALS =      5

 CPU TIME = 0:00:00.50
 DIRECT I/O COUNT = 6
 ELAPSED TIME = 00:00:01.47
```

FIGURE 45 GO1 DATA FOR GO MODEL DEPENDENCIES - EITHER PUMP TRAIN REQUIRED

```
GO2  (KSCGO, VAX VERSION 1.0)  RUN ON 20-OCT-82 AT 15:42:46
COPYRIGHT (C) 1982 by KAMAN SCIENCES CORPORATION

GO2 DATA FOR GO MODEL DEPENDENCIES - EITHER PUMP

PERF = 0, MXSIZE = 2048

OPERATOR FILE --- GO1 DATA FOR GO MODEL DEPENDENCIES - EITHER PUMP

RECORD KIND DATA
------ ------------------------------------------------------------
     1  1 5 2 0 0.95 1 0.05 $ WATER SOURCE
     2  2 5 2 0 0.99 1 0.01 $ POWER SOURCE
     3  3 6 0.9 0.1 0.0 $ PUMP
------------------------------------------------------------------

USE SUMMARY TABLE, ENTRY = KIND/TYPE(FREQUENCY)
   (FREQUENCY IS NEGATIVE FOR PERFECT KINDS.)

  1/ 5(     1)     2/ 5(     1)     3/ 6(     2)

NUMBER OF KINDS INPUT------    3
NUMBER USED - NONPERFECT--     3
NUMBER USED - PERFECT------    0

   1 FILE RECORDS WRITTEN FOR      5 OPERATORS.

 CPU TIME = 0:00:00.42
 DIRECT I/O COUNT = 13
 ELAPSED TIME = 00:00:01.12
```

FIGURE 46 GO2 DATA FOR GO MODEL DEPENDENCIES - EITHER PUMP TRAIN REQUIRED

```
GO3  (KSCGO, VAX VERSION 1.0)  RUN ON 20-OCT-82 AT 15:42:48
COPYRIGHT (C) 1982 by KAMAN SCIENCES CORPORATION

GO3 DATA FOR GO MODEL DEPENDENCIES - EITHER PUMP

OPERATOR FILE --- GO1 DATA FOR GO MODEL DEPENDENCIES - EITHER PUMP
KIND FILE ------- GO2 DATA FOR GO MODEL DEPENDENCIES - EITHER PUMP

RUN NUMBER  1
     PMIN =0.0000E+00
     NEW = 0, INTER = 0, SAVE = 0, MXDIST =  3000
     FIRST = 10000, LAST = 10000, TRACE = 2.00000

----------------------------------------
FINAL EVENT TABLE (INFINITY =   1)

                SIGNALS AND THEIR VALUES
                -------------------------
PROBABILITY       5
------------   ----
 0.0689050000     1
 0.9310950000     0

----------------------------------------

TOTAL PROBABILITY =  1.0000000000
TOTAL ERROR =        0.0000000000


INDIVIDUAL SIGNAL PROBABILITY DISTRIBUTIONS

VAL.         5
----   ------------
  0    0.9310950000
  1    0.0689050000


 CPU TIME = 0:00:00.35
 DIRECT I/O COUNT = 6
 ELAPSED TIME = 00:00:00.53
```

FIGURE 47 GO3 DATA FOR GO MODEL DEPENDENCIES - EITHER PUMP TRAIN REQUIRED

event 5_0, system success, occurs with probability 0.9311. This is the correct value, again establishing the validity of the KSC GO algorithms for handling dependent events.

The event tree for this revised model is shown in Figure 48. It is identical to the event tree of Figure 44 except for the combination of the branches in an OR gate, rather than an AND gate, in the final column.

These two models have demonstrated how dependent events are modeled and manipulated in the GO software to generate the correct probabilities of occurrence of all events.

Example 4 - Alarm System

This alarm system is somewhat more complex than the previous examples. In this example system, we introduce three values which the random variables may take. These values represent component and system operational responses of premature, success, and failure. The use of type 6 operators, the definition and use of a supertype, and the request for a final joint distribution involving three signals are also portrayed in this example.

Figure 49 depicts a quadruply redundant alarm system with two two-out-of-three detector and alarm systems receiving the sensor outputs. The two-out-of-three detectors are used to prevent false alarms caused by sensor malfunction -- transmitting a signal when the sensed event has not occurred. Presumably false alarms are a costly nuisance and the alarm system design attempts to preclude them by requiring that at least two sensors must give the same response to trigger an alarm. An alarm will be sounded if either of the two detector and alarm units respond, given the occurrence of the monitored event.

Figure 50 is the GO model developed for this system. Type 6 operators having three operational modes -- premature, success, and failure -- are used to represent the sensors. Supertype 150 is defined with three inputs and one output to model a two-out-of-three detector and alarm unit. The supertype is then used twice in the system model. Signals 10 and 20 are the signals for channels A and B, and signal 100 is the system output signal.

In this model we define three values which the signals may take. These are:

VALUE	DEFINITION
0	Premature operation; occurs before alarm event
1	Time at which alarm event occurs
2	Never

The printouts from GO1 and GO2 document the configuration and the data for the alarm system. They are provided in Figures 51 and 52. There are 15 operators in GO1, with a maximum number of five active signals. Six different kinds of component data are defined.

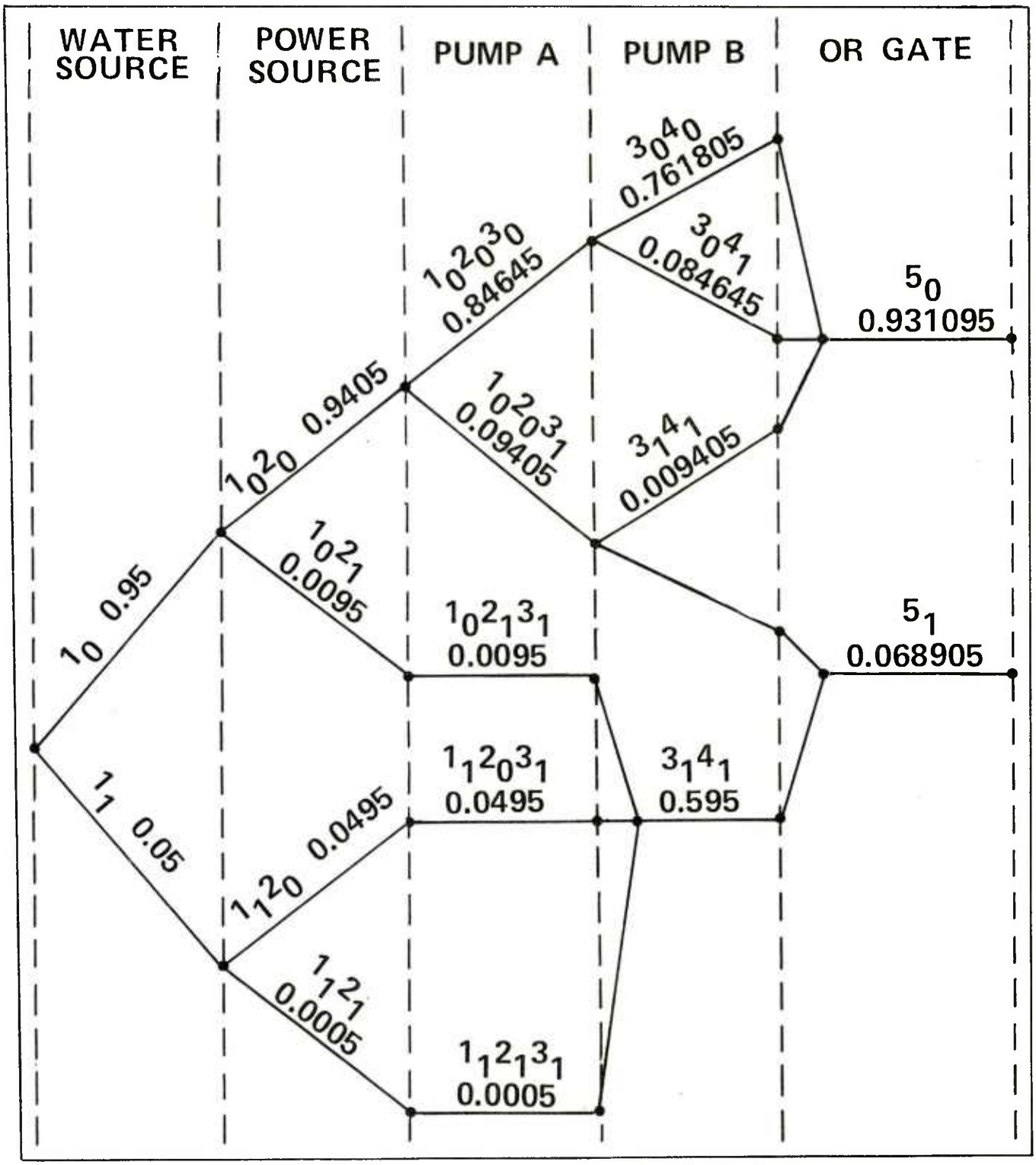


FIGURE 48 EVENT TREE GO MODEL DEPENDENCIES - EITHER PUMP TRAIN REQUIRED

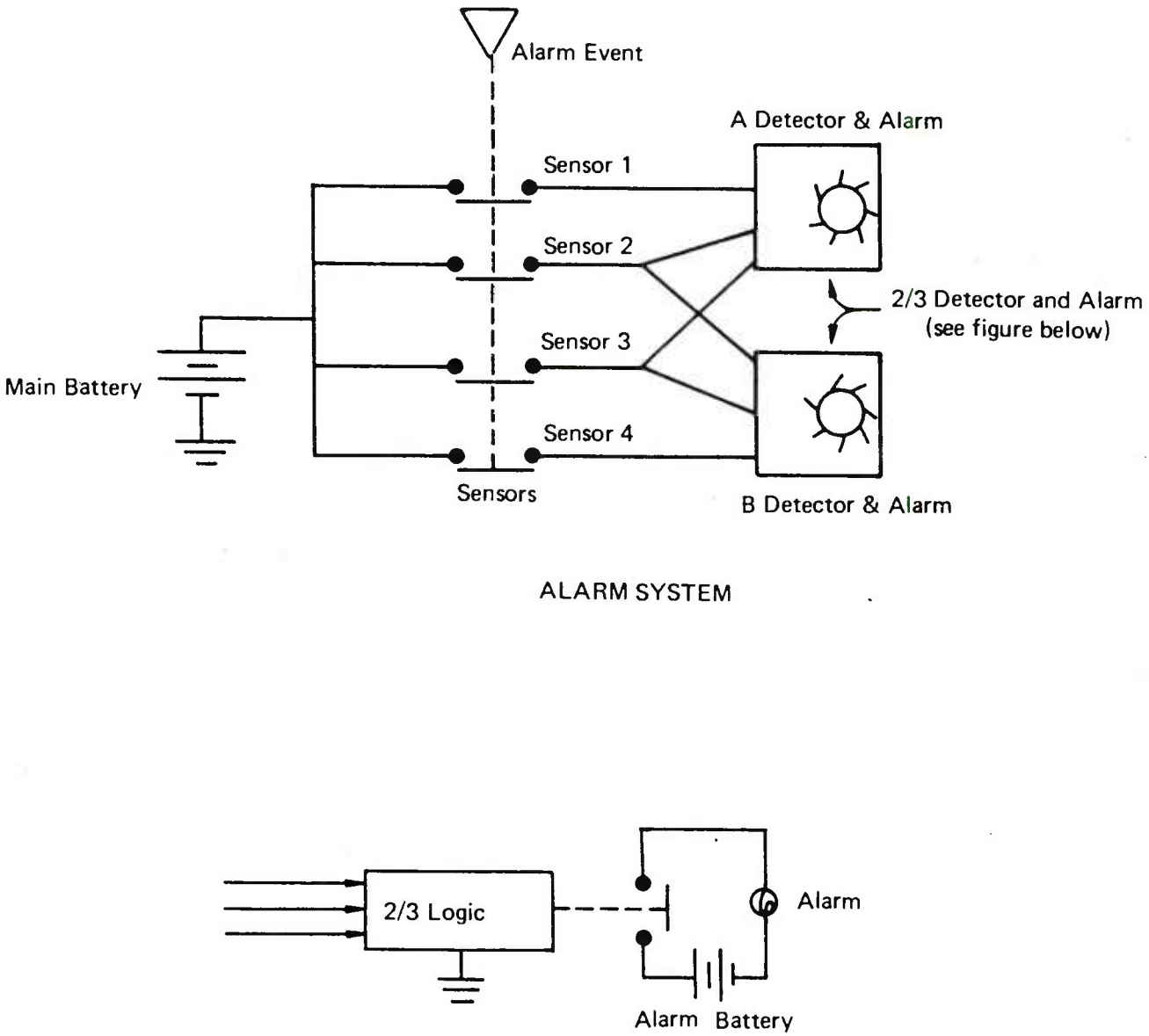


FIGURE 49 ALARM SYSTEM

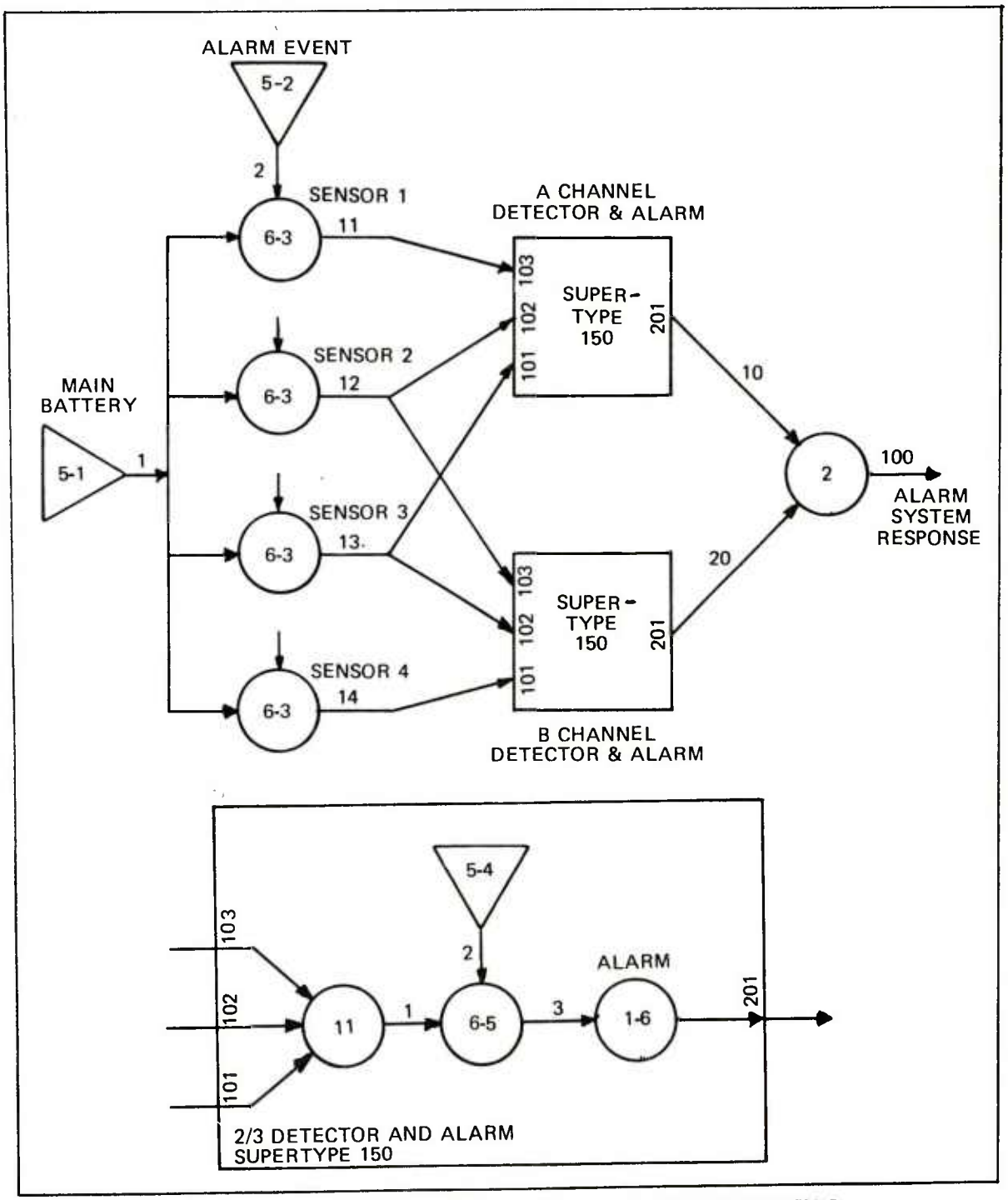


FIGURE 50 GO MODEL OF ALARM SYSTEM

```
GO1  (KSCGO, VAX VERSION 1.0)  RUN ON 22-OCT-82 AT 16:48:04
COPYRIGHT (C) 1982 by KAMAN SCIENCES CORPORATION

GO1 DATA FOR ALARM SYSTEM

INFIN =   2, VALUES =   3, BIAS =  750, OPS = 1; SIGNALS = 1; ERRORS = 25

 OP    DATA
----   ------------------------------------------------------------------
XXXX   150 -1 101 102 103 201 $ ST 150 DEFINITION
XXXX   11 2 3 101 102 103 1 $ 2 OF 3 GATE
XXXX   5 4 2 $ ALARM BATTERY
XXXX   6 5 1 2 3 $ SWITCH(NORMALLY OPEN)
XXXX   1 6 3 201 $ ALARM
   1   EUREKA
----   END OF SUPER TYPE 150
   1   5 1 1 $ MAIN BATTERY
   2   5 2 2 $ ALARM EVENT
   3   6 3 1 2 11 $ SENSOR 1
   4   6 3 1 2 12 $ SENSOR 2
   5   6 3 1 2 13 $ SENSOR 3
   6   6 3 1 2 14 $ SENSOR 4
$$$$   150 0 13 12 11 10 $ A DETECTOR AND ALARM
   7   (L=1)      11     2     3    13    12    11   751 $$ A DETECT2 OF 3 G
   8   (L=1)       5     4   752 $$ A DETECTALARM BA
   9   (L=1)       6     5   751   752   753 $$ A DETECTSWITCH(N
  10   (L=1)       1     6   753    10 $$ A DETECTALARM
$$$$   150 0 14 13 12 20 $ B DETECTOR AND ALARM
  11   (L=1)      11     2     3    14    13    12   754 $$ B DETECT2 OF 3 G
  12   (L=1)       5     4   755 $$ B DETECTALARM BA
  13   (L=1)       6     5   754   755   756 $$ B DETECTSWITCH(N
  14   (L=1)       1     6   756    20 $$ B DETECTALARM
  15   2 0 2 10 20 100 $ OR GATE
//16   0 10 20 100 $ SIGNALS FOR A CHANNEL, B CHANNEL, AND SYSTEM
```

FIGURE 51 GO1 OUTPUT FOR ALARM SYSTEM

```
SIGNAL DATA

          SOURCE OPER.
SIGNAL     NUM TYPE KIND   USING OPERATORS (- IF DELETED AT)
------ ------ ---- ----   ------------------------------------------------
     1      1    5      1      3      4      5     -6
     2      2    5      2      3      4      5     -6
    10     10    1      6     15
    11      3    6      3     -7
    12      4    6      3      7    -11
    13      5    6      3      7    -11
    14      6    6      3    -11
    20     14    1      6     15
   100     15    2      0
   751      7   11      2     -9
   752      8    5      4     -9
   753      9    6      5    -10
   754     11   11      2    -13
   755     12    5      4    -13
   756     13    6      5    -14

OPERATOR(NUMBER OF ACTIVE SIGNALS)
----------------------------------
    1(  1)    2(  2)    3(  3)    4(  4)    5(  5)    6(  4)    7(  4)    8(  5)
    9(  4)   10(  4)   11(  2)   12(  3)   13(  2)   14(  2)   15(  3)

NUMBER OF OPERATORS... =    15
NUMBER OF SIGNALS..... =    15
MAXIMUM NUMBER ACTIVE. =     5
MAX SIGNAL LIST SIZE.. =     5
NUMBER OF SIGNALS/WORD =    16
MAXIMUM WORDS/TERM.... =     1

FINAL SIGNALS =     10   20  100

 CPU TIME = 0:00:01.16
 DIRECT I/O COUNT = 10
 ELAPSED TIME = 00:00:02.66
```

FIGURE 51 (CONTINUED)

```
GO2  (KSCGO, VAX VERSION 1.0)  RUN ON 22-OCT-82 AT 16:48:08
COPYRIGHT (C) 1982 by KAMAN SCIENCES CORPORATION

GO2 DATA FOR ALARM SYSTEM

PERF = 0, MXSIZE = 2048

OPERATOR FILE --- GO1 DATA FOR ALARM SYSTEM

RECORD KIND DATA
------ ----------------------------------------------------------------------
     1  1 5 2 0 0.98 2 0.02 $ MAIN BATTERY
     2  2 5 1 1 1.0 $ ALARM EVENT
     3  3 6 0.97 0.02 0.01 $ SENSOR
     4  4 5 2 0 0.99 2 0.01 $ ALARM BATTERY
     5  5 6 0.990 0.007 0.003 $ SWITCH
     6  6 1 0.99 0.01 $ ALARM
------------------------------------------------------------------------------

USE SUMMARY TABLE, ENTRY = KIND/TYPE(FREQUENCY)
   (FREQUENCY IS NEGATIVE FOR PERFECT KINDS.)

  1/ 5(     1)     2/ 5(     1)     3/ 6(     4)     4/ 5(     2)     5/ 6(     2)
  6/ 1(     2)

NUMBER OF KINDS INPUT------    6
NUMBER USED - NONPERFECT--    6
NUMBER USED - PERFECT------    0

   1 FILE RECORDS WRITTEN FOR     15 OPERATORS.

 CPU TIME = 0:00:00.61
 DIRECT I/O COUNT = 14
 ELAPSED TIME = 00:00:01.76
```

FIGURE 52 GO2 OUTPUT FOR ALARM SYSTEM

The results obtained from executing this model are recorded in Figure 53. Referring to the portion of the figure entitled Individual Signal Probability Distributions, the output distributions for signals 10, 20, and 100, which represent channel A, channel B, and the system (either channel A or channel B), are shown.

The probability of a single channel (signal 10 or signal 20) functioning properly is 0.9524. The probability of failure on a single channel is 0.0473. The probability of a false alarm (premature) on a single channel is 3×10^{-4}.

The alarm system has premature, success, and failure probabilities of 5×10^{-4}, 0.9784, and 0.0211 respectively. Notice that the false alarm rate for the system is approximately double that for a single channel. The system failure rate is about half that for a single channel.

The Final Event Table of Figure 53 shows the complete joint distribution of signals 10, 20, and 100. The most likely event $10_1 20_1 100_1$, which occurs with probability 0.9260, is that both channels and the system are successful. All possible events expressed as the combinations of premature, success, and failure of each of the three signals are recorded with their respective probabilities of occurrence.

To conclude, this example has documented; (1) the use of three values representing premature, success, and failure operational modes; (2) the use of type 6 operators; (3) the use of supertypes; and (4) multiple output signals in the final joint distribution.

Example 5 - Two-of-Three Pump Trains

The purpose of this example is to portray the development of a GO model for a slightly more complex system, introduce the type 11 operator and use the model to simultaneously generate the availability of two different pump train designs.

The system to be analyzed is represented by Figure 54. A company presently has a pump system consisting of two pump trains (pumps, associated valves, and piping) and is considering a proposed system of three pump trains. The present system is representing by solid lines with the proposed additional pump train in dotted lines. What will be the increase in system availability with the three pump train system over the present system if two pumps must operate to provide proposed flow and pressure?

Because the three pump trains are logically identical we will represent a single train with supertype 500. Supertype 500 will then be called three times, once for each redundant train.

The GO model for this system is documented in Figure 55.

Because this is an availability model, the signals will take two values 0(available) and 1(unavailable). Of particular interest in the GO model for this system are the two ways in which we have combined the output signals from

```
GO3  (KSCGO, VAX VERSION 1.0)  RUN ON 22-OCT-82 AT 16:48:10
COPYRIGHT (C) 1982 by KAMAN SCIENCES CORPORATION

GO3 DATA FOR ALARM SYSTEM

OPERATOR FILE --- GO1 DATA FOR ALARM SYSTEM
KIND FILE ------- GO2 DATA FOR ALARM SYSTEM

RUN NUMBER  1
     PMIN =0.0000E+00
     NEW = 0, INTER = 0, SAVE = 0, MXDIST =  3000
     FIRST = 10000, LAST = 10000, TRACE = 2.00000

----------------------------------------
FINAL EVENT TABLE (INFINITY =   2)

                SIGNALS AND THEIR VALUES
                ------------------------
PROBABILITY       10   20  100
-------------   ---- ---- ----
 0.0000076721      0    2    0
 0.0000076721      2    0    0
 0.0000946688      0    0    0
 0.0001818926      0    1    0
 0.0001818926      1    0    0
 0.0211465091      2    2    2
 0.0261715023      2    1    1
 0.0261715023      1    2    1
 0.9260366882      1    1    1

----------------------------------------

TOTAL PROBABILITY =  1.0000000000
TOTAL ERROR =        0.0000000000


INDIVIDUAL SIGNAL PROBABILITY DISTRIBUTIONS

VAL.        10             20             100
----  -------------  -------------  -------------
  0   0.0002842335   0.0002842335   0.0004737982
  1   0.9523900830   0.9523900830   0.9783796927
  2   0.0473256835   0.0473256835   0.0211465091


 CPU TIME = 0:00:00.58
 DIRECT I/O COUNT = 7
 ELAPSED TIME = 00:00:01.22
```

FIGURE 53 GO3 OUTPUT FOR ALARM SYSTEM

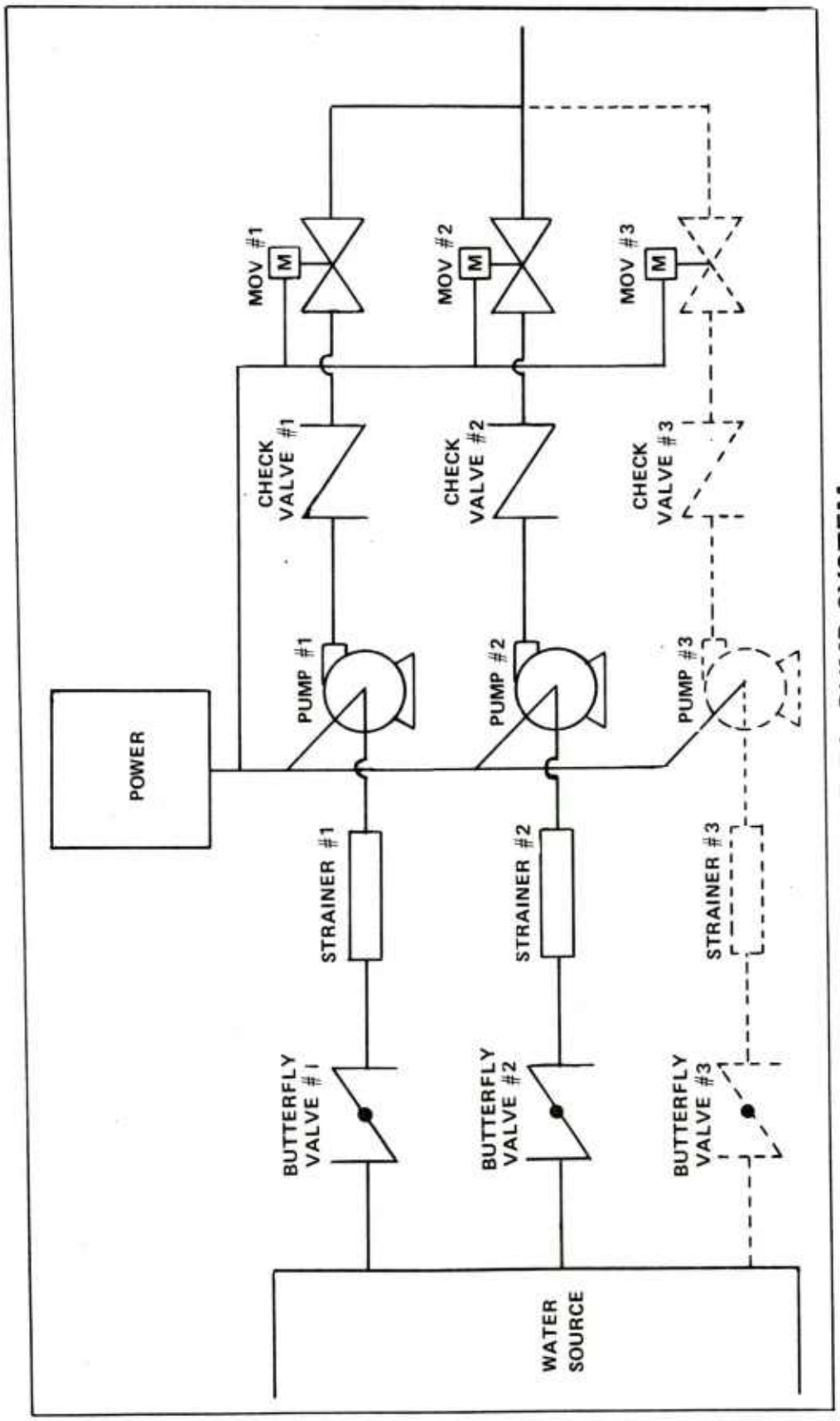


FIGURE 54 PUMP SYSTEM

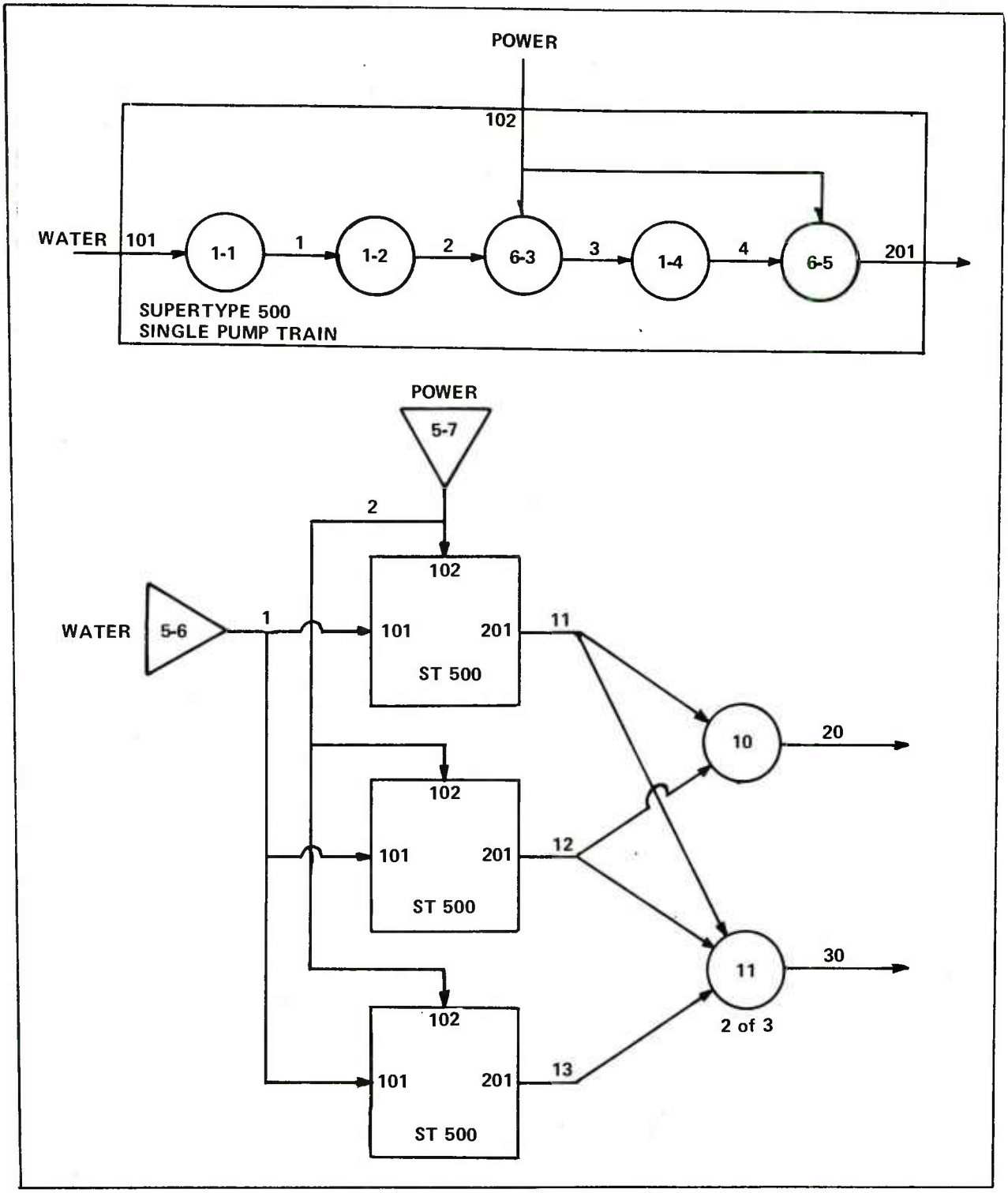


FIGURE 55 GO MODEL FOR PUMP SYSTEM

the pump trains. The type 10 "AND" gate which combines signals 11 and 12, the outputs from the two pump trains in the present system, generates signal 20. Signal 20 represents the present system.

In contrast, signal 30, the output from the type 11 operator which in this case is a 2-of-3 gate, represents the system availability for the proposed new system design.

The actual GO1 data developed for this model and input to the computer is portrayed in Figure 56. Notice that we first defined supertype 500 then used it thrice in the system model.

```
GO1 DATA FOR PUMP SYSTEM AVAILABILITY
 $PARAM INFIN=1 $
500 -1 101 102 201 $ PUMP TRAIN
1 1 101 1 $ BUTTERFLY VALVE
1 2 1 2 $ STRAINER
6 3 2 102 3 $ PUMP
1 4 3 4 $ CHECK VALVE
6 5 4 102 201 $ MOV
END
5 6 1 $ WATER SOURCE
5 7 2 $ POWER SOURCE
500 0 1 2 11 $ PUMP TRAIN 1
500 0 1 2 12 $ PUMP TRAIN 2
500 0 1 2 13 $ PUMP TRAIN 3
10 0 2 11 12 20 $ TWO TRAINS
11 2 3 11 12 13 30 $ TWO OF THREE TRAINS
0 20 30 $ FINAL SIGNALS
```

FIGURE 56 GO1 INPUT DATA FOR PUMP SYSTEM AVAILABILITY

The GO1 output for this system model is recorded in Figure 57. The model consists of 19 supertypes. Signals 20 and 30 are the final output signals.

The GO2 component availability data for this model are actually input to the computer on a GO2 data input file as recorded in Figure 58. For this system it was assumed that the component availabilities would not change in the 2-of-3 system over the 2-of-2 system.

```
GO2 DATA FOR PUMP SYSTEM AVAILABILITY
 $PARAM $
1 1 0.9995 0.0005 $ BUTTERFLY VALVE
2 1  0.9990 0.0010 $ STRAINER
3 6 0.995 0.005 0.0 $ PUMP
4 1 0.999 0.001 $ CHECK VALVE
5 6 0.992 0.008 0.0 $ MOV
6 5 2 0 0.999 1 0.001 $ WATER
7 5 2 0 0.9999 1 0.0001 $ POWER
END
```

FIGURE 58 GO2 INPUT DATA FOR SYSTEM AVAILABILITY

The GO2 output data for this model is recorded in Figure 58.

The two lines of GO3 input data for this model are:

```
GO3 DATA FOR PUMP SYSTEM AVAILABILITY
 $PARAM PMIN=0.0$
```

```
G01  (KSCGO, VAX VERSION 1.0)  RUN ON 26-OCT-82 AT 11:27:53
COPYRIGHT (C) 1982 by KAMAN SCIENCES CORPORATION

G01 DATA FOR PUMP SYSTEM AVAILABILITY

INFIN =   1, VALUES =   2, BIAS =  750, OPS = 1, SIGNALS = 1,
                                                 ERRORS = 25
 OP    DATA
----   ------------------------------------------------------
XXXX   500 -1 101 102 201 $ PUMP TRAIN
XXXX   1 1 101 1 $ VALVE-BY
XXXX   1 2 1 2 $ STRAINER
XXXX   6 3 2 102 3 $ PUMP
XXXX   1 4 3 4 $ VALV-CHK
XXXX   6 5 4 102 201 $ MOV
   1   END
----   END OF SUPER TYPE 500
   1   5 6 1 $ WATER SOURCE
   2   5 7 2 $ POWER SOURCE
$$$$   500 0 1 2 11 $ TRAIN 1
   3   (L=1)       1     1     1   751 $$ TRAIN 1 VALVE-BY
   4   (L=1)       1     2   751   752 $$ TRAIN 1 STRAINER
   5   (L=1)       6     3   752     2   753 $$ TRAIN 1 PUMP
   6   (L=1)       1     4   753   754 $$ TRAIN 1 VALV-CHK
   7   (L=1)       6     5   754     2    11 $$ TRAIN 1 MOV
$$$$   500 0 1 2 12 $ TRAIN 2
   8   (L=1)       1     1     1   755 $$ TRAIN 2 VALVE-BY
   9   (L=1)       1     2   755   756 $$ TRAIN 2 STRAINER
  10   (L=1)       6     3   756     2   757 $$ TRAIN 2 PUMP
  11   (L=1)       1     4   757   758 $$ TRAIN 2 VALV-CHK
  12   (L=1)       6     5   758     2    12 $$ TRAIN 2 MOV
$$$$   500 0 1 2 13 $ TRAIN 3
  13   (L=1)       1     1     1   759 $$ TRAIN 3 VALVE-BY
  14   (L=1)       1     2   759   760 $$ TRAIN 3 STRAINER
  15   (L=1)       6     3   760     2   761 $$ TRAIN 3 PUMP
  16   (L=1)       1     4   761   762 $$ TRAIN 3 VALV-CHK
  17   (L=1)       6     5   762     2    13 $$ TRAIN 3 MOV
  18   10 0 2 11 12 20 $ TWO TRAINS
  19   11 2 3 11 12 13 30 $ TWO OF THREE TRAINS
//20   0 20 30 $ FINAL SIGNALS
```

FIGURE 57 G01 DATA FOR PUMP SYSTEM AVAILABILITY

```
SIGNAL DATA

          SOURCE OPER.
SIGNAL    NUM TYPE KIND   USING OPERATORS (- IF DELETED AT)
------ ---- ---- ----   ----------------------------------------------------
     1    1    5    6      3     8   -13
     2    2    5    7      5     7    10    12    15   -17
    11    7    6    5     18   -19
    12   12    6    5     18   -19
    13   17    6    5    -19
    20   18   10    0
    30   19   11    2
   751    3    1    1     -4
   752    4    1    2     -5
   753    5    6    3     -6
   754    6    1    4     -7
   755    8    1    1     -9
   756    9    1    2    -10
   757   10    6    3    -11
   758   11    1    4    -12
   759   13    1    1    -14
   760   14    1    2    -15
   761   15    6    3    -16
   762   16    1    4    -17

OPERATOR(NUMBER OF ACTIVE SIGNALS)
-----------------------------------
    1(  1)    2(  2)    3(  3)    4(  3)    5(  3)    6(  3)    7(  3)    8(  4)
    9(  4)   10(  4)   11(  4)   12(  4)   13(  4)   14(  4)   15(  4)   16(  4)
   17(  3)   18(  4)   19(  2)

NUMBER OF OPERATORS... =    19
NUMBER OF SIGNALS..... =    19
MAXIMUM NUMBER ACTIVE. =     4
MAX SIGNAL LIST SIZE.. =     4
NUMBER OF SIGNALS/WORD =    32
MAXIMUM WORDS/TERM.... =     1

FINAL SIGNALS =    20   30

 CPU TIME = 0:00:01.15
 DIRECT I/O COUNT = 11
 ELAPSED TIME = 00:00:02.47
```

FIGURE 57 (CONTINUED)

```
GO2  (KSCGO, VAX VERSION 1.0)  RUN ON 26-OCT-82 AT 10:56:57
COPYRIGHT (C) 1982 by KAMAN SCIENCES CORPORATION

GO2 DATA FOR PUMP SYSTEM AVAILABILITY

PERF = 0, MXSIZE = 2048

OPERATOR FILE --- GO1 DATA FOR PUMP SYSTEM AVAILABILITY

RECORD KIND DATA
------ ----------------------------------------------------------------
     1  1 1 0.9995 0.0005 $ BUTTERFLY VALVE
     2  2 1  0.9990 0.0010 $ STRAINER
     3  3 6 0.995 0.005 0.0 $ PUMP
     4  4 1 0.999 0.001 $ CHECK VALVE
     5  5 6 0.992 0.008 0.0 $ MOV
     6  6 5 2 0 0.999 1 0.001 $ WATER
     7  7 5 2 0 0.9999 1 0.0001 $ POWER
     8  END
-------------------------------------------------------------------------

USE SUMMARY TABLE. ENTRY = KIND/TYPE(FREQUENCY)
   (FREQUENCY IS NEGATIVE FOR PERFECT KINDS.)

  1/ 1(     3)     2/ 1(     3)     3/ 6(     3)     4/ 1(     3)     5/ 6(     3)
  6/ 5(     1)     7/ 5(     1)

NUMBER OF KINDS INPUT-----    7
NUMBER USED - NONPERFECT--    7
NUMBER USED - PERFECT-----    0

   1 FILE RECORDS WRITTEN FOR    19 OPERATORS.

 CPU TIME = 0:00:00.65
 DIRECT I/O COUNT = 14
 ELAPSED TIME = 00:00:02.11
```

FIGURE 59 GO2 DATA FOR PUMP SYSTEM AVAILABILITY

The GO3 output data is shown in its entirety in Figure 60. The Final Event Table of Figure 60 records the joint distributions of signals 20 and 30 reflecting the system availability for the two pump train configurations.

The Individual Signal Probability Distributions document that the availability of the present system two pump trains is 0.9683 (event 20_0). The proposed new 2-of-3 pump system configuration will increase availability to 0.9982 (event 30_0).

Being able to quantify the performance of this proposed design alteration quickly provides the facts necessary to determine the economic benefits and penalties associated with its proposal. A capable GO analyst can develop the GO model of a system of this complexity and execute it in less than one hour's time.

This example has introduced the type 11 operators and demonstrated the use of the KSC GO methodology to simultaneously generate system availability for two different system configurations.

Example 6 - Weapon Fuzing System

The purpose of this example is to demonstrate the application of the GO methodology to analyze the safety and reliability of a generic weapon fuzing system. This example portrays the use of a number of values which the signals may take to represent premature weapon operation (a serious safety problem), success, late operation, and failure. The system is sufficiently complex that an equation for its operation cannot be written by inspection necessitating the use of the GO methodology.

The fuzing system to be analyzed is shown schematically in Figure 61. The parallel pairs of normally open contacts are safety and sequential fuzing blocks which are to be removed at different times to send proper electrical signals to arm and fire the weapon (not shown). Success will be achieved if the following combination of output signals, properly sequenced in time, are received:

$$(C_A \cup C_B) \cap (D_A \cup D_B) \cap (E_A \cup E_B).$$

One might think of signals C_A and C_B as pre-arm signals, D_A and D_B as final-arm signals, and E_A and E_B as the fire signals. The subscripts A and B refer to the channel. Fuzing success as the equation above indicates requires both a pre-arm and a final-arm signal on either channel and a fire signal from either channel.

The sequenced outputs are generated by the times of arrival of external signals W, battery initiation, X_A and X_B, Y and Z. The expected sequence of operation is (1) the receipt of signal W of time 1 energizing squib switches S_{1_A} and S_{1_B} closing the six pairs of normally open switch contacts on all output lines; (2) actuation of batteries U_A and U_B at time 1 simultaneous with the receipt of signal W; (3) function of

```
GO3  (KSCGO, VAX VERSION 1.0)  RUN ON 26-OCT-82 AT 10:57:00
COPYRIGHT (C) 1982 by KAMAN SCIENCES CORPORATION

GO3 DATA FOR PUMP SYSTEM AVAILABILITY

OPERATOR FILE --- GO1 DATA FOR PUMP SYSTEM AVAILABILITY
KIND FILE ------- GO2 DATA FOR PUMP SYSTEM AVAILABILITY

RUN NUMBER  1
     PMIN =0.0000E+00
     NEW = 0, INTER = 0, SAVE = 0, MXDIST =  3000
     FIRST = 10000, LAST = 10000, TRACE = 2.00000

----------------------------------------
FINAL EVENT TABLE (INFINITY =   1)

               SIGNALS AND THEIR VALUES
               ------------------------
PROBABILITY      20   30
-------------  ---- ----
 0.0018056317     1    1
 0.0298738996     1    0
 0.9683204687     0    0

----------------------------------------

TOTAL PROBABILITY =  1.0000000000
TOTAL ERROR =        0.0000000000


INDIVIDUAL SIGNAL PROBABILITY DISTRIBUTIONS

VAL.         20             30
----  ------------  ------------
  0   0.9683204687  0.9981943683
  1   0.0316795313  0.0018056317


 CPU TIME = 0:00:00.35
 DIRECT I/O COUNT = 6
 ELAPSED TIME = 00:00:01.72
```

FIGURE 60 GO3 DATA FOR PUMP SYSTEM AVAILABILITY

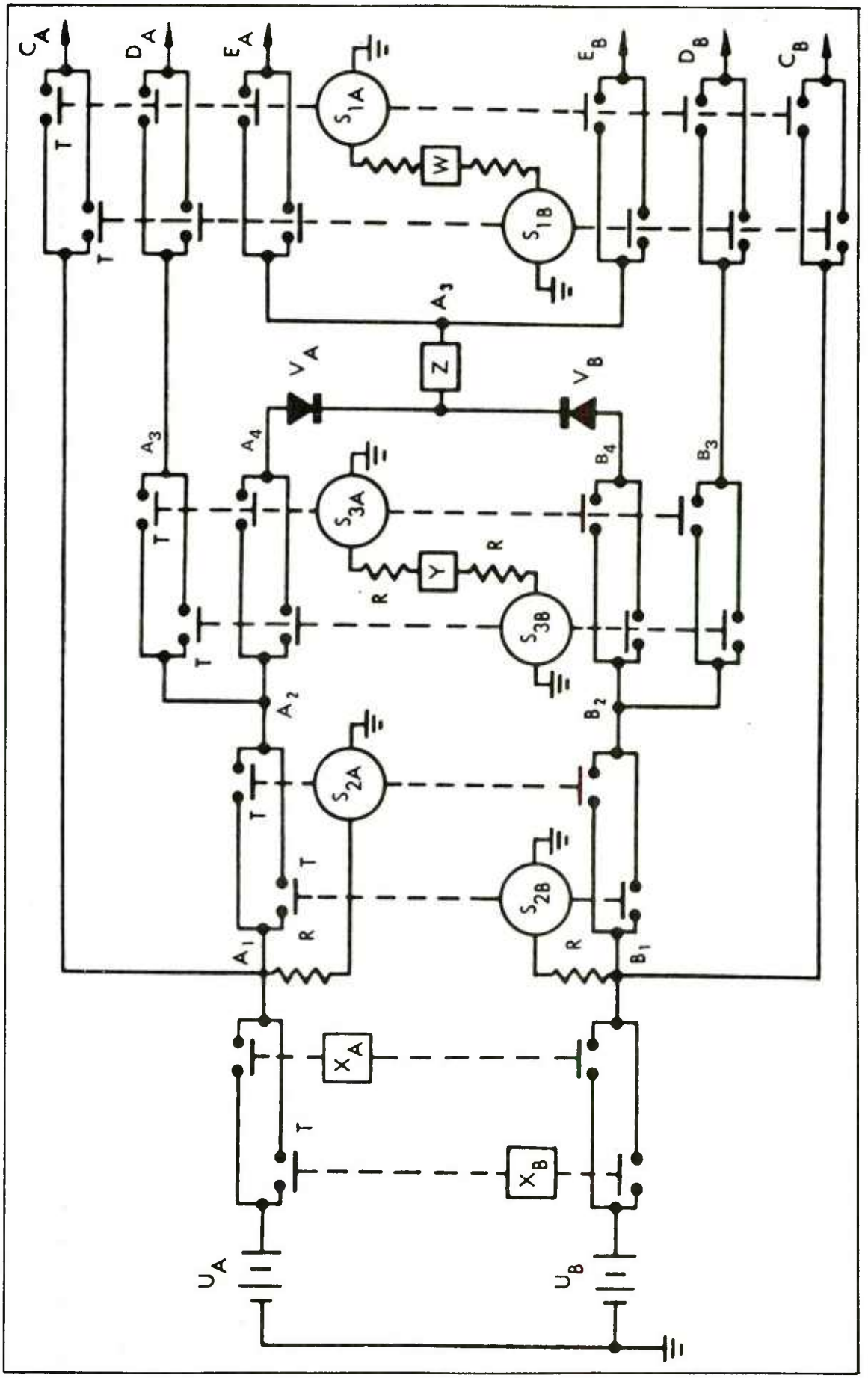


FIGURE 61 WEAPON FUZING SYSTEM

accelerometers X_A and X_B; (4) the receipt of signal Y; (5) the receipt of signal Z. Figure 62 defines the values which the signals may take in the GO model of this system.

VALUE	DEFINITION
0	All time prior to arrival of signal W.
1	Time instant at which signal W should normally arrive.
2	Time instant between the arrival of signal W, X_A, and X_B.
3	Time instant at which signal X_A and X_B should normally arrive.
4	Time instant between arrival of signals X_A, X_B, and Y.
5	Time instant at which signal Y should normally arrive.
6	Time instant between arrival of signals Y and Z.
7	Time instant at which signal Z should normally arrive.
8	Final time point of interest.
15	Never (i.e., at infinite time).

FIGURE 62 WEAPON FUZING SYSTEM SIGNAL VALUE DEFINITION AND TIME REFERENCE

A GO model of the weapon fuzing system is shown in Figure 63. Supertype 100 was defined as shown in Figure 64 to represent the ubiquitous pairs of normally open switch contacts. Signal 45 is the final output signal. A reliable fuzing system is represented as event 45_7, signal 45 taking value 7. The events 45_4, 45_5, or 45_6, represent premature weapon detonations and are safety problems of varying severity. Event 45_8 represents a late detonation. The event 45_{15} represents the failure of the fuzing system.

Figure 65 depicts the GO1 output for this system model. The numerous uses of supertype 100 are recorded as is the system configuration. Seventy-two operators are used. Signal 45 is the only output signal requested.

Figure 66 records the probability input data for each component class (kind). The probability mass functions showing the probabilities of arrival of the external input signals W, X, Y, and Z are shown. Data for the kind 12 type 9 operator representing the sprytron which requires a precise combination of the times of arrival of its two input signals is also shown.

The GO3 output data of Figure 67 records that there are six possible system events. The most likely event, 45_7, indicates a system reliability of 0.993547. The probability of system failure, 45_{15}, is 0.005291. The probabilities of occurrence of the three premature events range from $1x10^{-6}$ to $9.9x10^{-5}$. The late detonation event, 45_8, occurs with probability $1x10^{-3}$.

This example has portrayed the application of the GO methodology to simultaneously analyze both the safety and the reliability of a generic weapon fuzing system. The dependent effects of all component operational modes and all external input signals are combined and used to characterize weapon performance. The dependent effects simultaneously influencing both safety and reliability are properly incorporated.

Example 7 - Communication Network Analysis

The purpose of this example is to demonstrate the application of the KSC GO methodology to analyze the probability of information flow in a simple communication network. The communication network to be analyzed is one treated by J. deMercado, et. al., in IEEE Transactions on Reliability, Volume R-25, June 1976, pp. 71-76.

The network is shown in Figure 68. The object of the analysis is to determine the probability of transmitting information from node s to node t given the existing links between nodes and the probabilities of proper information transmission on the links. Note that the direction of information flow is shown with arrowheads. On all links except ac and bc the flow is uni-directional. In this system the nodes are perfect.

The network is transformed into the block diagram of Figure 69 with the links shown as blocks and the nodes as circles.

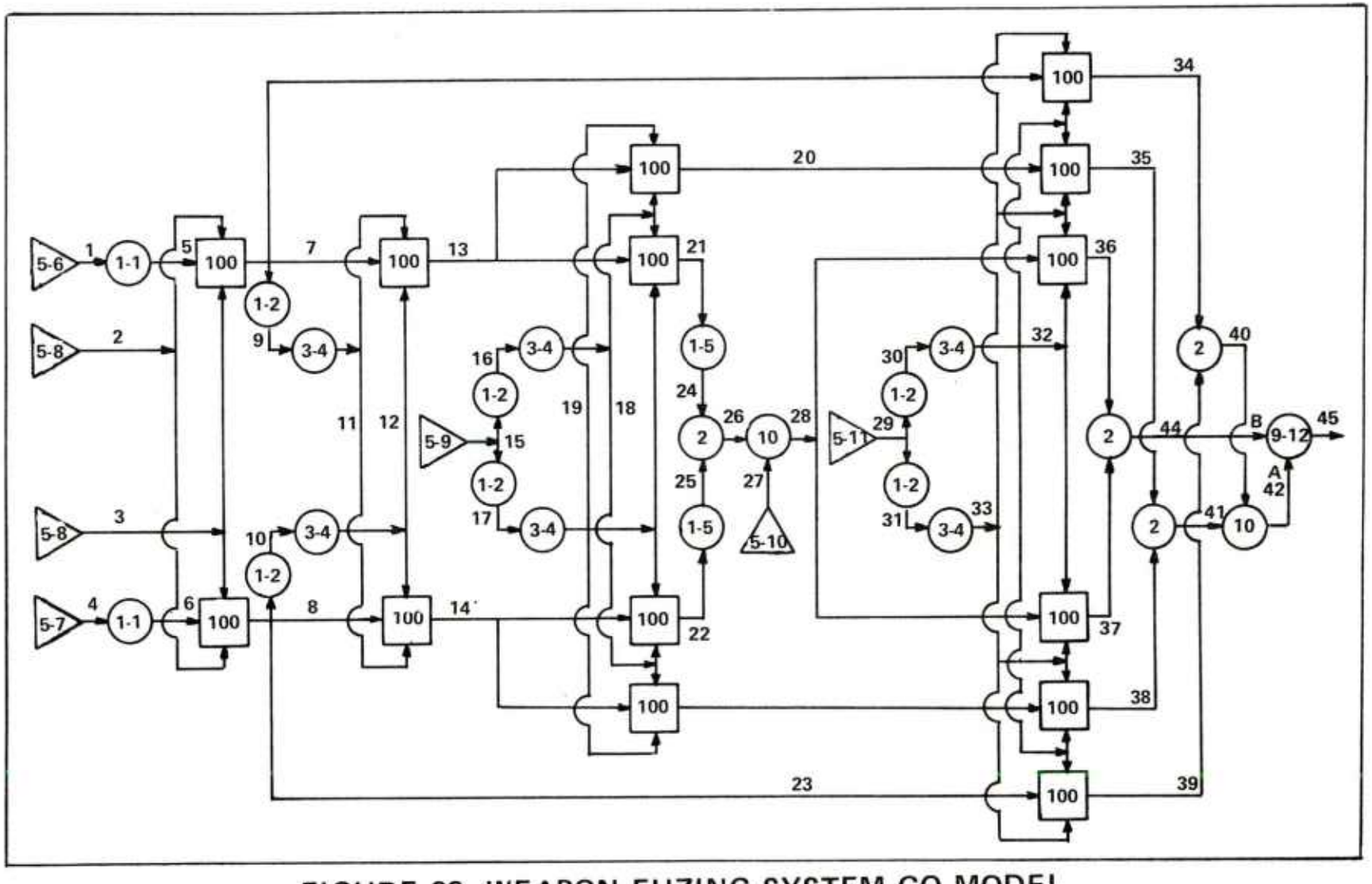


FIGURE 63 WEAPON FUZING SYSTEM GO MODEL

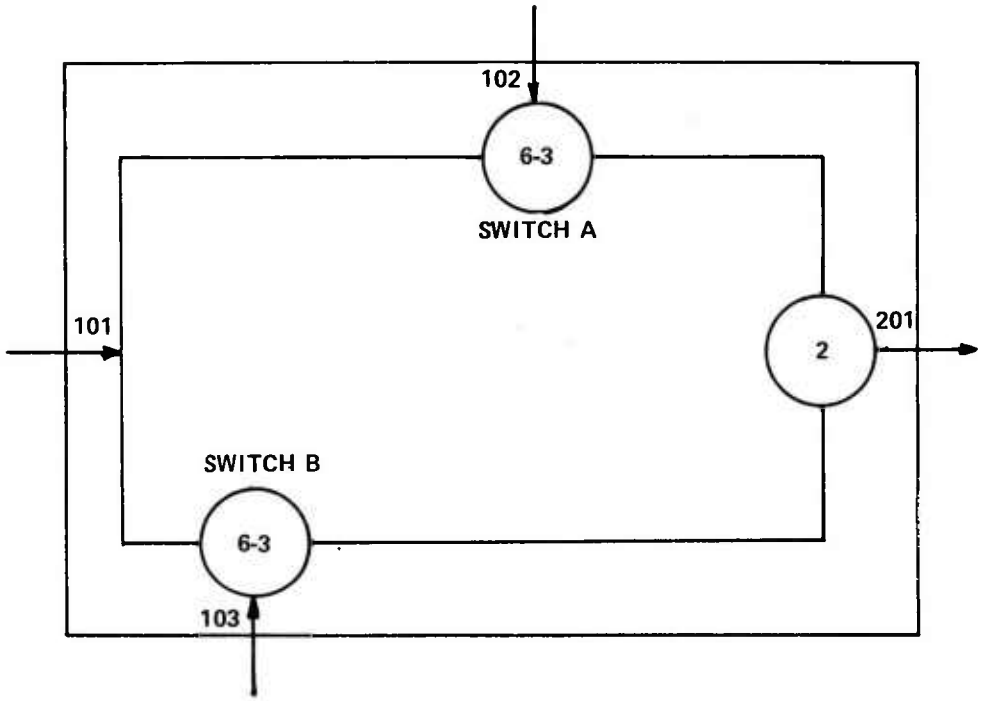


FIGURE 64 SUPERTYPE REPRESENTING PAIR OF NORMALLY OPEN SWITCH CONTACTS

```
GO1  (KSCGO, VAX VERSION 1.0)  RUN ON  1-NOV-82 AT 16:07:49
COPYRIGHT (C) 1982 by KAMAN SCIENCES CORPORATION

EX3 - GO1

INFIN =  15, VALUES =  16, BIAS =  750, OPS = 1, SIGNALS = 1,
                                                 ERRORS = 25

 OP    DATA
-----  ------------------------------------------------------------
XXXX   100 -1 101 102 103 201 1001 $
XXXX   6 1001 101 102 1 $ SWITCH A
XXXX   6 1001 101 103 2 $ SWITCH B
XXXX   2 0 2 1 2 201 $ OR GATE
   1   END $
----   END OF SUPER TYPE 100
   1   5 6 1 $ START A
   2   1 1 1 5 $ BATTERY A
   3   5 7 4 $ START B
   4   5 8 2 $ INPUT X-A
   5   5 8 3 $ INPUT X-B
   6   1 1 4 6 $ BATTERY B
$$$$   100 0 5 2 3 7 3 $ A1
   7   (L=1)       6     3     5     2   751 $$ A1       SWITCH A
   8   (L=1)       6     3     5     3   752 $$ A1       SWITCH B
   9   (L=1)       2     0     2   751   752     7 $$ A1       OR GATE
$$$$   100 0 6 2 3 8 3 $ B1
  10   (L=1)       6     3     6     2   753 $$ B1       SWITCH A
  11   (L=1)       6     3     6     3   754 $$ B1       SWITCH B
  12   (L=1)       2     0     2   753   754     8 $$ B1       OR GATE
  13   1 2 7 9 $ RESISTOR A1
  14   3 4 9 11 $ S2A
  15   1 2 8 10 $ RESISTOR B1
  16   3 4 10 12 $ S2B
$$$$   100 0 7 11 12 13 3 $ A2
  17   (L=1)       6     3     7    11   755 $$ A2       SWITCH A
  18   (L=1)       6     3     7    12   756 $$ A2       SWITCH B
  19   (L=1)       2     0     2   755   756    13 $$ A2       OR GATE
$$$$   100 0 8 11 12 14 3 $ B2
  20   (L=1)       6     3     8    11   757 $$ B2       SWITCH A
  21   (L=1)       6     3     8    12   758 $$ B2       SWITCH B
  22   (L=1)       2     0     2   757   758    14 $$ B2       OR GATE
  23   5 9 15 $ INPUT Y
  24   1 2 15 16 $ RESISTOR A2
  25   1 2 15 17 $ RESISTOR B2
  26   3 4 16 18 $ S3A
  27   3 4 17 19 $ S3B
$$$$   100 0 13 18 19 20 3 $ A3
  28   (L=1)       6     3    13    18   759 $$ A3       SWITCH A
  29   (L=1)       6     3    13    19   760 $$ A3       SWITCH B
  30   (L=1)       2     0     2   759   760    20 $$ A3       OR GATE
```

FIGURE 65 GO1 OUTPUT FOR WEAPON FUZING SYSTEM

```
$$30  100 0 13 18 19 21 3 $ A4
  31  (L=1)       6    3    13    18   761 $$ A4       SWITCH A
  32  (L=1)       6    3    13    19   762 $$ A4       SWITCH B
  33  (L=1)       2    0     2   761   762    21 $$ A4       OR GATE
$$34  100 0 14 18 19 22 3 $ B4
  34  (L=1)       6    3    14    18   763 $$ B4       SWITCH A
  35  (L=1)       6    3    14    19   764 $$ B4       SWITCH B
  36  (L=1)       2    0     2   763   764    22 $$ B4       OR GATE
$$37  100 0 14 18 19 23 3 $ B3
  37  (L=1)       6    3    14    18   765 $$ B3       SWITCH A
  38  (L=1)       6    3    14    19   766 $$ B3       SWITCH B
  39  (L=1)       2    0     2   765   766    23 $$ B3       OR GATE
  40  1 5 21 24 $ V-A
  41  1 5 22 25 $ V-B
  42  2 0 2 24 25 26 $ OR GATE
  43  5 10 27 $ INPUT Z
  44  10 0 2 26 27 28 $ AND GATE
  45  5 11 29 $ INPUT W
  46  1 2 29 30 $ RESISTOR A3
  47  1 2 29 31 $ RESISTOR B3
  48  3 4 30 32 $ S1A
  49  3 4 31 33 $ S1B
$$50  100 0 7 32 33 34 3 $ CA
  50  (L=1)       6    3     7    32   767 $$ CA       SWITCH A
  51  (L=1)       6    3     7    33   768 $$ CA       SWITCH B
  52  (L=1)       2    0     2   767   768    34 $$ CA       OR GATE
$$53  100 0 20 32 33 35 3 $ DA
  53  (L=1)       6    3    20    32   769 $$ DA       SWITCH A
  54  (L=1)       6    3    20    33   770 $$ DA       SWITCH B
  55  (L=1)       2    0     2   769   770    35 $$ DA       OR GATE
$$56  100 0 28 32 33 36 3 $ EA
  56  (L=1)       6    3    28    32   771 $$ EA       SWITCH A
  57  (L=1)       6    3    28    33   772 $$ EA       SWITCH B
  58  (L=1)       2    0     2   771   772    36 $$ EA       OR GATE
$$59  100 0 8 32 33 39 3 $ CB
  59  (L=1)       6    3     8    32   773 $$ CB       SWITCH A
  60  (L=1)       6    3     8    33   774 $$ CB       SWITCH B
  61  (L=1)       2    0     2   773   774    39 $$ CB       OR GATE
$$62  100 0 23 32 33 38 3 $ DB
  62  (L=1)       6    3    23    32   775 $$ DB       SWITCH A
  63  (L=1)       6    3    23    33   776 $$ DB       SWITCH B
  64  (L=1)       2    0     2   775   776    38 $$ DB       OR GATE
$$65  100 0 28 32 33 37 3 $ EB
  65  (L=1)       6    3    28    32   777 $$ EB       SWITCH A
  66  (L=1)       6    3    28    33   778 $$ EB       SWITCH B
  67  (L=1)       2    0     2   777   778    37 $$ EB       OR GATE
  68  2 0 2 36 37 44 $ FIRE
  69  10 0 2 34 35 40 $ CA AND DA
  70  10 0 2 38 39 41 $ CB AND DB
  71  2 0 2 40 41 42 $ ARM
  72  9 12 42 44 45 $ SPRYTRON
//73  0 45 $ FINAL SIGNAL
```

FIGURE 65 (CONTINUED)

SIGNAL DATA

SIGNAL	SOURCE NUM	OPER. TYPE	KIND	USING OPERATORS (- IF DELETED AT)					
1	1	5	6	-2					
2	4	5	8	7	-10				
3	5	5	8	8	-11				
4	3	5	7	-6					
5	2	1	1	7	-8				
6	6	1	1	10	-11				
7	9	2	0	13	17	18	50	-51	
8	12	2	0	15	20	21	59	-60	
9	13	1	2	-14					
10	15	1	2	-16					
11	14	3	4	17	-20				
12	16	3	4	18	-21				
13	19	2	0	28	29	31	-32		
14	22	2	0	34	35	37	-38		
15	23	5	9	24	-25				
16	24	1	2	-26					
17	25	1	2	-27					
18	26	3	4	28	31	34	-37		
19	27	3	4	29	32	35	-38		
20	30	2	0	53	-54				
21	33	2	0	-40					
22	36	2	0	-41					
23	39	2	0	62	-63				
24	40	1	5	-42					
25	41	1	5	-42					
26	42	2	0	-44					
27	43	5	10	-44					
28	44	10	0	56	57	65	-66		
29	45	5	11	46	-47				
30	46	1	2	-48					
31	47	1	2	-49					
32	48	3	4	50	53	56	59	62	-65
33	49	3	4	51	54	57	60	63	-66
34	52	2	0	-69					
35	55	2	0	-69					
36	58	2	0	-68					
37	67	2	0	-68					
38	64	2	0	-70					
39	61	2	0	-70					
40	69	10	0	-71					
41	70	10	0	-71					
42	71	2	0	-72					
44	68	2	0	-72					
45	72	9	12						

FIGURE 65 (CONTINUED)

```
  751      7    6       3     -9
  752      8    6       3     -9
  753     10    6       3    -12
  754     11    6       3    -12
  755     17    6       3    -19
  756     18    6       3    -19
  757     20    6       3    -22
  758     21    6       3    -22
  759     28    6       3    -30
  760     29    6       3    -30
  761     31    6       3    -33
  762     32    6       3    -33
  763     34    6       3    -36
  764     35    6       3    -36
  765     37    6       3    -39
  766     38    6       3    -39
  767     50    6       3    -52
  768     51    6       3    -52
  769     53    6       3    -55
  770     54    6       3    -55
  771     56    6       3    -58
  772     57    6       3    -58
  773     59    6       3    -61
  774     60    6       3    -61
  775     62    6       3    -64
  776     63    6       3    -64
  777     65    6       3    -67
  778     66    6       3    -67
```

```
NUMBER OF OPERATORS... =    72
NUMBER OF SIGNALS..... =    72
MAXIMUM NUMBER ACTIVE. =     9
MAX SIGNAL LIST SIZE.. =     9
NUMBER OF SIGNALS/WORD =     8
MAXIMUM WORDS/TERM.... =     2

FINAL SIGNALS =     45

 CPU TIME = 0:00:03.31
 DIRECT I/O COUNT = 25
 ELAPSED TIME = 00:00:04.74
```

```
OPERATOR(NUMBER OF ACTIVE SIGNALS)
----------------------------------
   1(  1)    2(  1)    3(  2)    4(  3)    5(  4)    6(  4)    7(  5)    8(  5)
   9(  4)   10(  4)   11(  3)   12(  2)   13(  3)   14(  3)   15(  4)   16(  4)
  17(  5)   18(  6)   19(  5)   20(  5)   21(  5)   22(  4)   23(  5)   24(  6)
  25(  6)   26(  6)   27(  6)   28(  7)   29(  8)   30(  7)   31(  8)   32(  8)
  33(  7)   34(  8)   35(  9)   36(  8)   37(  8)   38(  7)   39(  6)   40(  6)
  41(  6)   42(  5)   43(  6)   44(  5)   45(  6)   46(  7)   47(  7)   48(  7)
  49(  7)   50(  8)   51(  8)   52(  7)   53(  8)   54(  8)   55(  7)   56(  8)
  57(  9)   58(  8)   59(  9)   60(  9)   61(  8)   62(  9)   63(  9)   64(  8)
  65(  8)   66(  7)   67(  6)   68(  5)   69(  4)   70(  3)   71(  2)   72(  1)
```

FIGURE 65 (CONTINUED)

```
GO2  (KSCGO, VAX VERSION 1.0)  RUN ON  1-NOV-82 AT 16:07:55
COPYRIGHT (C) 1982 by KAMAN SCIENCES CORPORATION

EX3 - GO2

PERF = 0, MXSIZE = 2048

OPERATOR FILE --- EX3 - GO1

RECORD KIND DATA
------ ------------------------------------------------------------------------
     1  1 1 .968 .032 $ BATTERY
     2  2 1 1 0 $ RESISTOR
     3  3 6 .988 .01 .002 $ NORMALLY OPEN SWITCH CONTACTS
     4  4 3 .9899 .0031 .007 $ SWITCH ACTUATOR
     5  5 1 .988 .012 $ DIODE
     6  6 5 1 1 1 $ START A
     7  7 5 1 1 1 $ START B
     8  8 5 5 0 .0005 2 .001 3 .99 4 .0005 8 .008 $ INPUT X
     9  9 5 9 0 .0002 1 .001 2 .0005 3 .0001 4 .001 5 .996 6 .0001 7 .0001 8 .001 $ INPUT Y
    10  10 5 7 1 .0002 2 .0001 4 .0001 5 .0005 6 .0001 7 .998 8 .001 $ INPUT Z
    11  11 5 3 0 .0005 1 .999 8 .0005 $ INPUT W
    12  12 9 7 0 15 1 1 2 2 3 3 4 4 5 5 6 6 $ SPRYTRON
------------------------------------------------------------------------

USE SUMMARY TABLE. ENTRY = KIND/TYPE(FREQUENCY)
   (FREQUENCY IS NEGATIVE FOR PERFECT KINDS.)

  1/ 1(     2)     2/ 1(     6)     3/ 6(    28)     4/ 3(     6)     5/ 1(     2)
  6/ 5(     1)     7/ 5(     1)     8/ 5(     2)     9/ 5(     1)    10/ 5(     1)
 11/ 5(     1)    12/ 9(     1)

NUMBER OF KINDS INPUT-----  12
NUMBER USED - NONPERFECT--  12
NUMBER USED - PERFECT-----   0

   2 FILE RECORDS WRITTEN FOR    72 OPERATORS.

 CPU TIME = 0:00:01.12
 DIRECT I/O COUNT = 20
 ELAPSED TIME = 00:00:02.33
```

FIGURE 66 GO2 OUTPUT FOR WEAPON FUZING SYSTEM

```
GO3  (KSCGO, VAX VERSION 1.0)  RUN ON  1-NOV-82 AT 16:07:58
COPYRIGHT (C) 1982 by KAMAN SCIENCES CORPORATION

EX3 - GO3

OPERATOR FILE --- EX3 - GO1
KIND FILE ------- EX3 - GO2

RUN NUMBER  1
     PMIN =1.0000E-08
     NEW = 0, INTER = 1, SAVE = 0, MXDIST =  3000
     FIRST = 10000, LAST = 10000, TRACE = 2.00000
----------------------------------------
FINAL EVENT TABLE (INFINITY =  15)

               SIGNALS AND THEIR VALUES
               ------------------------

PROBABILITY      45
-------------   ----
 0.0000014703     4
 0.0000145555     5
 0.0000986118     6
 0.0010040127     8
 0.0052914723    15
 0.9935471604     7

----------------------------------------

TOTAL PROBABILITY =  0.9999572829
TOTAL ERROR =        0.0000427171


INDIVIDUAL SIGNAL PROBABILITY DISTRIBUTIONS

VAL.        45
----   ------------
  4    0.0000014703
  5    0.0000145555
  6    0.0000986118
  7    0.9935471604
  8    0.0010040127
 15    0.0052914723


 CPU TIME = 0:00:11.31
 DIRECT I/O COUNT = 18
 ELAPSED TIME = 00:00:11.99
```

FIGURE 67 GO3 OUTPUT FOR WEAPON FUZING SYSTEM

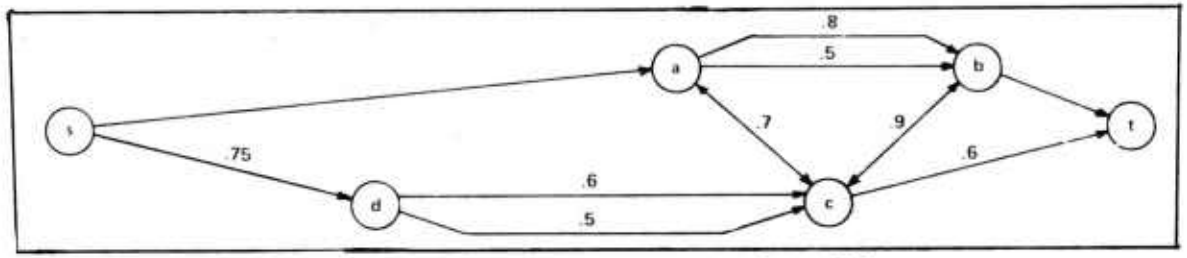


FIGURE 68 NETWORK GRAPH

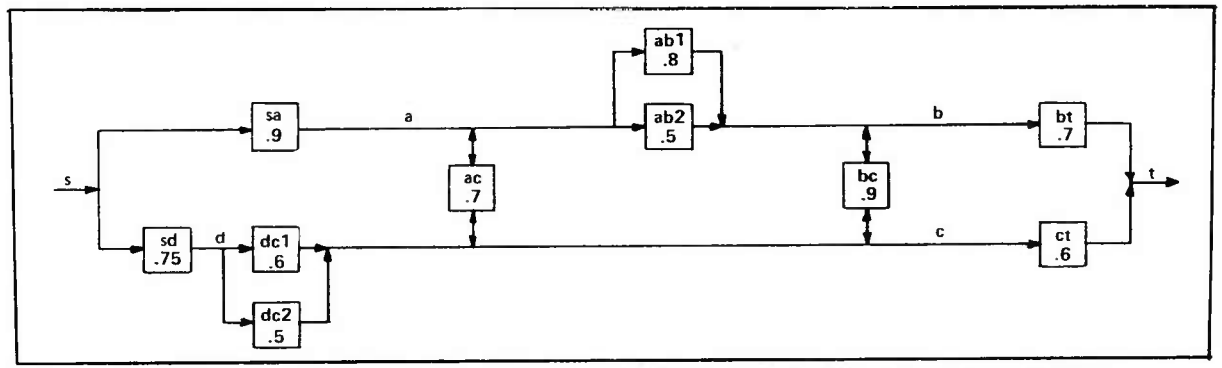


FIGURE 69 BLOCK DIAGRAM OF NETWORK GRAPH

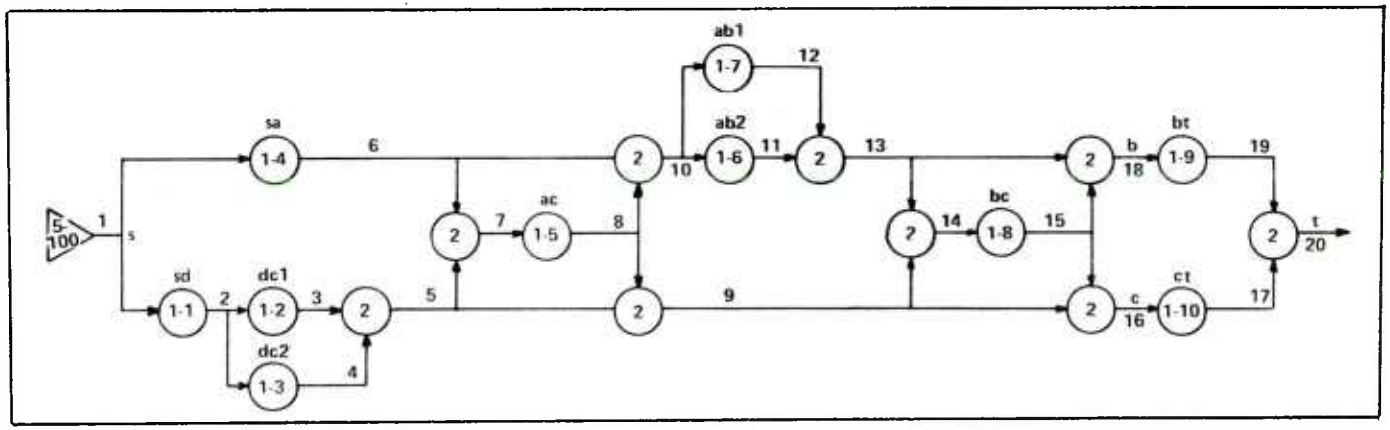


FIGURE 70 GO CHART OF NETWORK GRAPH

We now develop a GO model of the network using the standard symbols. The GO model is shown as Figure 70. Figures 68, 69, and 70, are shown on the same page to show the relationship of the GO model to the network and its block diagram.

The outputs from programs GO1, GO2, and GO3, shown in Figures 71, 72, and 73, document the network model, the transmission probability data, and the system transmission probability. The probability of successfully transmitting information from node s to node t with the link probabilities specified is 0.8307072.

More sophisticated treatment of communication systems involving node uncertainties and additional two-way transmission links is discussed in the KSC GO Model Development Manual.

```
GO1  (KSCGO, VAX VERSION 1.0)  RUN ON  5-NOV-82 AT 08:23:10
COPYRIGHT (C) 1982 by KAMAN SCIENCES CORPORATION

GO1 COMMUNICATION NETWORK ANALYSIS

INFIN =   1, VALUES =   2, BIAS =  750, OPS = 1, SIGNALS = 1,
                                                 ERRORS = 25

 OP    DATA
----   ------------------------------------------------------------
   1   5 100 1 $ START
   2   1 1 1 2 $ SD
   3   1 2 2 3 $ DC1
   4   1 3 2 4 $ DC2
   5   2 0 2 3 4 5 $ OR GATE
   6   1 4 1 6 $ SA
   7   2 0 2 5 6 7 $ OR GATE
   8   1 5 7 8 $ AC
   9   2 0 2 5 8 9 $ OR GAT
  10   2 0 2 6 8 10 $ OR GATE
  11   1 7 10 12 $ AB1
  12   1 6 10 11 $ AB2
  13   2 0 2 11 12 13 $ OR GATE
  14   2 0 2 9 13 14 $ OR GATE
  15   1 8 14 15 $ BC
  16   2 0 2 9 15 16 $ OR GATE
  17   2 0 2 13 15 18 $ OR GATE
  18   1 10 16 17 $ CT
  19   1 9 18 19 $ BT
  20   2 0 2 17 19 20 $ OR GATE
//ZZ   0 20 $ FINAL SIGNAL
```

FIGURE 71 GO1 COMMUNICATION NETWORK ANALYSIS

```
SIGNAL DATA

          SOURCE OPER.
SIGNAL    NUM TYPE KIND   USING OPERATORS (- IF DELETED AT)
------ ---- ---- ----  ----------------------------------------------------
     1      1    5  100      2     -6
     2      2    1    1      3     -4
     3      3    1    2     -5
     4      4    1    3     -5
     5      5    2    0      7     -9
     6      6    1    4      7    -10
     7      7    2    0     -8
     8      8    1    5      9    -10
     9      9    2    0     14    -16
    10     10    2    0     11    -12
    11     12    1    6    -13
    12     11    1    7    -13
    13     13    2    0     14    -17
    14     14    2    0    -15
    15     15    1    8     16    -17
    16     16    2    0    -18
    17     18    1   10    -20
    18     17    2    0    -19
    19     19    1    9    -20
    20     20    2    0

OPERATOR(NUMBER OF ACTIVE SIGNALS)
----------------------------------
   1(  1)    2(  2)    3(  3)    4(  3)    5(  2)    6(  2)    7(  3)    8(  3)
   9(  3)   10(  2)   11(  3)   12(  3)   13(  2)   14(  3)   15(  3)   16(  3)
  17(  2)   18(  2)   19(  2)   20(  1)

NUMBER OF OPERATORS... =   20
NUMBER OF SIGNALS..... =   20
MAXIMUM NUMBER ACTIVE. =    3
MAX SIGNAL LIST SIZE.. =    3
NUMBER OF SIGNALS/WORD =   32
MAXIMUM WORDS/TERM.... =    1

FINAL SIGNALS =    20

                                CPU TIME = 0:00:01.39
                                DIRECT I/O COUNT = 10
                                ELAPSED TIME = 00:00:02.99
```

FIGURE 71 (CONTINUED)

```
GO2  (KSCGO, VAX VERSION 1.0)  RUN ON  5-NOV-82 AT 08:23:15
COPYRIGHT (C) 1982 by KAMAN SCIENCES CORPORATION

GO2 COMMUNICATION NETWORK ANALYSIS

PERF = 0, MXSIZE = 2048

OPERATOR FILE --- GO1 COMMUNICATION NETWORK ANALYSIS

RECORD KIND DATA
------ ----------------------------------------------------------------
     1   1 1 0.75 0.25 $ SD
     2   2 1 0.60 0.40 $ DC1
     3   3 1 0.50 0.50 $ DC2
     4   4 1 0.90 0.10 $ SA
     5   5 1 0.70 0.30 $ AC
     6   6 1 0.50 0.50 $ AB2
     7   7 1 0.80 0.20 $ AB1
     8   8 1 0.90 0.10 $ BC
     9   9 1 0.70 0.30 $ BT
    10   10 1 0.60 0.40 $ CT
    11   100 5 1 0 1, $ START
-------------------------------------------------------------------------

USE SUMMARY TABLE, ENTRY = KIND/TYPE(FREQUENCY)
   (FREQUENCY IS NEGATIVE FOR PERFECT KINDS.)

  1/ 1(     1)     2/ 1(     1)     3/ 1(     1)     4/ 1(     1)     5/ 1(     1)
  6/ 1(     1)     7/ 1(     1)     8/ 1(     1)     9/ 1(     1)    10/ 1(     1)
100/ 5(     1)

NUMBER OF KINDS INPUT-----   11
NUMBER USED - NONPERFECT--   11
NUMBER USED - PERFECT-----    0

   1 FILE RECORDS WRITTEN FOR    20 OPERATORS.

 CPU TIME = 0:00:00.77
 DIRECT I/O COUNT = 15
 ELAPSED TIME = 00:00:03.57
```

FIGURE 72 GO2 COMMUNICATION NETWORK ANALYSIS

```
GO3  (KSCGO, VAX VERSION 1.0)  RUN ON  5-NOV-82 AT 08:23:20
COPYRIGHT (C) 1982 by KAMAN SCIENCES CORPORATION

GO3 COMMUNICATION NETWORK ANALYSIS

OPERATOR FILE --- GO1 COMMUNICATION NETWORK ANALYSIS
KIND FILE ------- GO2 COMMUNICATION NETWORK ANALYSIS

RUN NUMBER  1
     PMIN =0.0000E+00
     NEW = 0, INTER = 0, SAVE = 0, MXDIST =  3000
     FIRST = 10000, LAST = 10000, TRACE = 2.00000

----------------------------------------
FINAL EVENT TABLE (INFINITY =   1)

                SIGNALS AND THEIR VALUES
                ------------------------
PROBABILITY       20
-------------   -----
 0.1692928000     1
 0.8307072000     0

----------------------------------------

TOTAL PROBABILITY =  1.0000000000
TOTAL ERROR =        0.0000000000


INDIVIDUAL SIGNAL PROBABILITY DISTRIBUTIONS

VAL.        20
----   ------------
  0    0.8307072000
  1    0.1692928000


 CPU TIME = 0:00:00.39
 DIRECT I/O COUNT = 6
 ELAPSED TIME = 00:00:01.11
```

FIGURE 73 GO3 COMMUNICATION NETWORK ANALYSIS

CHAPTER 5
FAULT FINDER

Introduction

The KSC Fault Finder programs are auxiliary to the basic KSC GO programs defined and exemplified in earlier chapters. The purpose of the Fault Finder is to answer the question, "What causes such and such an event to occur?" Usually the referenced event is a system failure which is caused by the failures of single components or the simultaneous failures of two or more components. Often there are many such failure combinations which could cause a selected system failure event to occur. The Fault Finder programs explicitly identify and manipulate these fault sets.

The term "fault set" was deliberately selected to reference these component failure combinations causing system failures, rather than "cut sets", because of the inherently more general nature of a GO model over a fault tree model. Components modeled using the GO methodology are not restricted to binary states, e.g., good, bad, as are the elements of a fault tree. Consequently to differentiate between the more general faults which might be defined for the operational states of components in a GO model and those of the more restricted two-state formulation of a fault tree, we have chosen the term fault set, rather than cut set, to describe the component or element failure combinations which cause system failures.

Fault Finder Example

The particular event -- known as the "selected event" -- is a term, or the logical union of several terms, of the final distribution produced by GO3. The selected event is defined by the user. The selected event in most cases is a system failure, but it may be any GO3 output event or combination of such events.

To introduce you to the Fault Finder capability we will use the model of the alarm system previously discussed as example 4 in Chapter 4. We reproduce here for convenience the GO3 output of the alarm system GO model as Figure 74. There are two system failure events. These are 100_2, representing an alarm system failure, and 100_0, representing an alarm system premature. We will execute the Fault Finder to find the component failure modes which cause each of these events to occur. A separate run will be made for each case. The selected event in the first case will be 100_2 entered in the form

```
100 2 $
```

```
GO3  (KSCGO, VAX VERSION 1.0)  RUN ON 22-OCT-82 AT 16:48:10
COPYRIGHT (C) 1982 by KAMAN SCIENCES CORPORATION

GO3 DATA FOR ALARM SYSTEM

OPERATOR FILE --- GO1 DATA FOR ALARM SYSTEM
KIND FILE ------- GO2 DATA FOR ALARM SYSTEM

RUN NUMBER  1
     PMIN =0.0000E+00
     NEW = 0, INTER = 0, SAVE = 0, MXDIST =  3000
     FIRST = 10000, LAST = 10000, TRACE = 2.00000

----------------------------------------
FINAL EVENT TABLE (INFINITY =   2)

                SIGNALS AND THEIR VALUES
                ------------------------
PROBABILITY       10    20   100
-------------   ----  ----  ----
 0.0000076721      0     2     0
 0.0000076721      2     0     0
 0.0000946688      0     0     0
 0.0001818926      0     1     0
 0.0001818926      1     0     0
 0.0211465091      2     2     2
 0.0261715023      2     1     1
 0.0261715023      1     2     1
 0.9260366882      1     1     1

----------------------------------------

TOTAL PROBABILITY =   1.0000000000
TOTAL ERROR =         0.0000000000


INDIVIDUAL SIGNAL PROBABILITY DISTRIBUTIONS

VAL.        10            20            100
----   ------------  ------------  ------------
  0    0.0002842335  0.0002842335  0.0004737902
  1    0.9523900830  0.9523900830  0.9783796927
  2    0.0473256835  0.0473256835  0.0211465091


 CPU TIME = 0:00:00.58
 DIRECT I/O COUNT = 7
 ELAPSED TIME = 00:00:01.22
```

FIGURE 74 GO3 OUTPUT FOR ALARM SYSTEM

in the input data file for program FF1. In the second case the selected event will be

100 0 $.

Before proceeding we define the use of critical mode codes for each GO operator. These are the use of numbers 0, 1, 2, and 3, to define the operational states of operators. The standard critical mode code definitions are shown at the top of Figure 75.

Most GO operators are given default values for their operational states as shown in the bottom half of Figure 75. For example a type 1 operator has two operational states, good and bad. Normally the good state is considered a success and is given a critical mode code of 0. Similarly the bad state is usually considered a failure and is given a critical mode code of 1. If this is what the user intends, no data need be entered because default mode codes of 0 and 1 have been entered for these two operational states. If the user wants to enter the mode codes explicitly, or desires that they be different than the default mode codes, this is done in the manner shown in Figure 75. For a type 1 operator, the data in FF2 is entered as

K 1 M_1 M_2 $

where M_1 and M_2 are the mode codes for the two operational states of the type 1 operator. Only the codes 0, 1, 2, and 3 are permissable. The default values for the type 1 are M_1=0, M_2=1.

Operators of type 2, 9, 10, 11, 14, 15, do not require critical mode code definitions because their operation is logically perfect -- i.e., they represent and perform logical, rather than functional, operations.

Figure 76 documents that the selected event -- the system failure event for which fault sets are sought -- is 100_2. The FF1 manipulation then alters the distribution file formerly created by GO3 with the parameter SAVE=1, eliminating all terms which do not lead to the selected event. In the process it also deletes perfect operators from consideration.

Now program FF2 is executed to find the fault sets by their order, i.e., the number of components whose simultaneous failure cause the selected event. The user has the option to override the default critical mode code values. In particular, data to specify the failure modes for type 5 operators is almost always required.

The output data for the FF2 run to find the fault sets causing system failure event 100_2 is shown in Figure 77. The user-defined critical mode codes for kind 1 type 5 and kind 4 type 5 are shown in essentially the same manner as input in GO2. The line entry

1 5 2 0 1 $

0 -- SUCCESS MODE

1 -- FAILURE MODE

2 -- PREMATURE MODE

3 -- VARIABLE MODE

KIND	TYPE*	CRITICAL MODE CODE	DEFAULTS
K	1	M_1 M_2	0 1
K	3	M_1 M_2 M_3	0 1 2
K	4	m $M_1 \ldots M_m$	$0 \ldots 0_m$
K	5	n $M_1 \ldots M_n$	$0 \ldots 0_n$
K	6	M_1 M_2 M_3	0 1 2
K	7	M_1 M_2 M_3	0 1 2
K	8	n $M_1 \ldots M_n$	$0 \ldots 0_n$
K	12	n $M_1 \ldots M_n$	------
K	13	n $M_1 \ldots M_n$	------

*TYPES 2, 9, 10, 11, 14, & 15 unnecessary.

FIGURE 75
FF2 CRITICAL MODE DEFINITION DATA INPUT SUMMARY

```
FF1 INPUT DATA FOR ALARM SYSTEM

MODEL: G01 DATA FOR ALARM SYSTEM
KINDS: G02 DATA FOR ALARM SYSTEM

INTER=0,  PRUNE=0.000000E+00

SELECTED EVENT - SIGNAL NUMBER(VALUE)
   1.    100(  2)

   4 PERFECT OPS DELETED
        2     7    11    15

  11 OPS LEFT IN TREE
        1     3     4     5     6     8     9    10    12    13

       14

   1 OP RECORDS IN FAULT FILE (FILE 40)
```

FIGURE 76 FF1 OUTPUT FOR ALARM SYSTEM SELECTED EVENT 100_2

```
FF2 DATA FOR ALARM SYSTEM

MODEL: G01 DATA FOR ALARM SYSTEM
KINDS: G02 DATA FOR ALARM SYSTEM

SELECTED EVENTS: SIGNAL NUMBER(VALUE)
   1.   100(  2)

INTER= 1,  FIRST= 0,  LAST= 4,     11 OPS

USER-DEFINED CRITICAL MODES

1 5 2 0 1 $
4 5 2 0 1 $

----- START OF 0-ORDER FAULTS -----

TREE VANISHED AT OP  10.  NO 0-ORDER FAULTS.

----- END OF 0-ORDER FAULTS -----


----- START OF 1-ORDER FAULTS -----

1-ORDER FAULT SETS
------------------
    SET  OP(MODE)   NAME
   ----- ----------------
     1.      1(1) MAIN BAT


----- END OF 1-ORDER FAULTS -----
```

FIGURE 77 FF2 OUTPUT FAULT SETS CAUSING SYSTEM FAILURE EVENT 100_2

----- START OF 2-ORDER FAULTS -----

2-ORDER FAULT SETS

SET	OP(MODE)	NAME	OP(MODE)	NAME
1.	4(1)	SENSOR 2	5(1)	SENSOR 3
2.	9(1)	A DESWIT	13(1)	B DESWIT
3.	9(1)	A DESWIT	14(1)	B DEALAR
4.	9(1)	A DESWIT	12(1)	B DEALAR
5.	10(1)	A DEALAR	13(1)	B DESWIT
6.	10(1)	A DEALAR	14(1)	B DEALAR
7.	10(1)	A DEALAR	12(1)	B DEALAR
8.	8(1)	A DEALAR	13(1)	B DESWIT
9.	8(1)	A DEALAR	14(1)	B DEALAR
10.	8(1)	A DEALAR	12(1)	B DEALAR

----- END OF 2-ORDER FAULTS -----

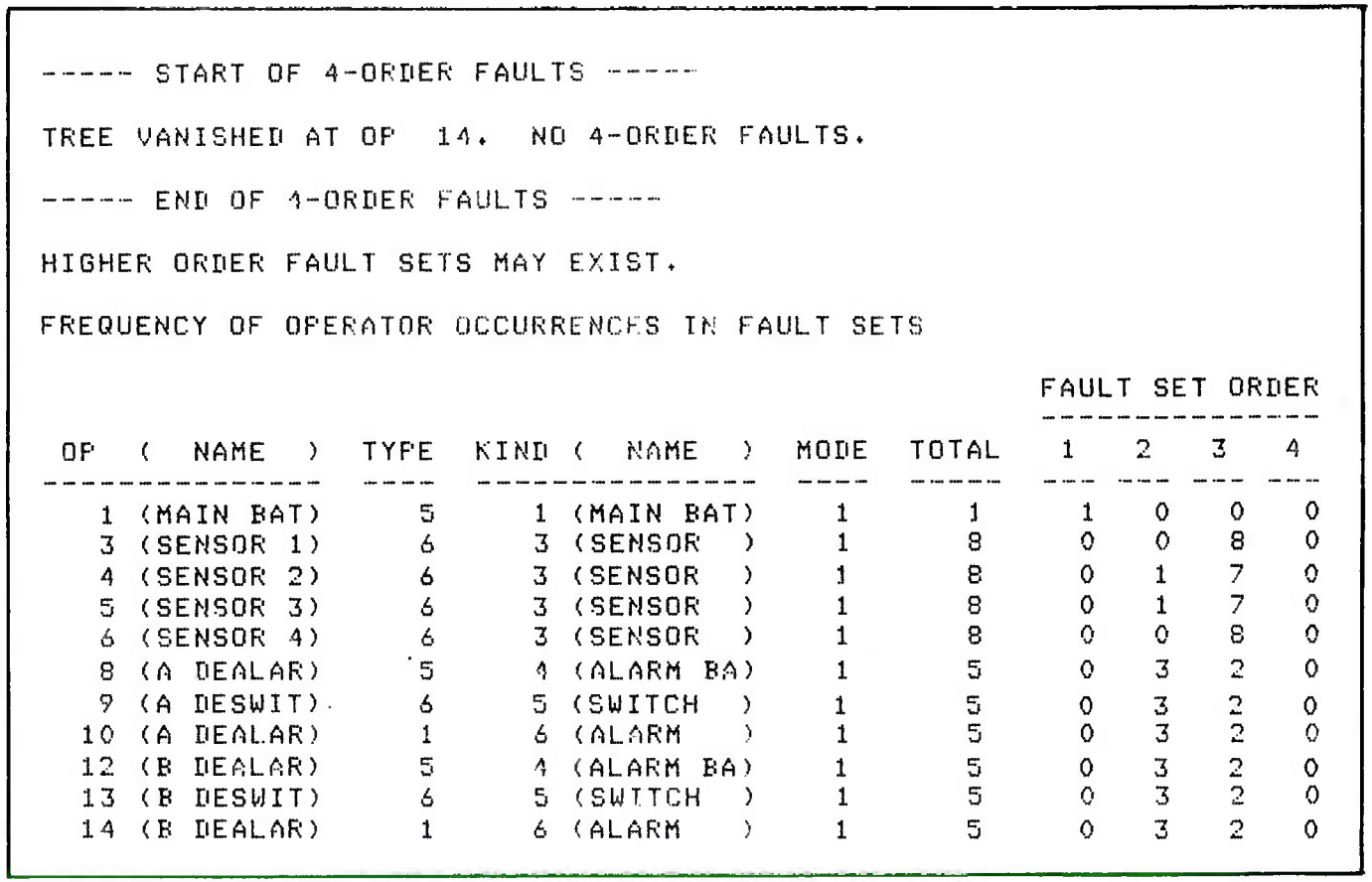
----- START OF 3-ORDER FAULTS -----

3-ORDER FAULT SETS

SET	OP(MODE)	NAME	OP(MODE)	NAME	OP(MODE)	NAME
1.	3(1)	SENSOR 1	4(1)	SENSOR 2	6(1)	SENSOR 4
2.	3(1)	SENSOR 1	4(1)	SENSOR 2	13(1)	B DESWIT
3.	3(1)	SENSOR 1	4(1)	SENSOR 2	14(1)	B DEALAR
4.	3(1)	SENSOR 1	4(1)	SENSOR 2	12(1)	B DEALAR
5.	3(1)	SENSOR 1	5(1)	SENSOR 3	6(1)	SENSOR 4
6.	3(1)	SENSOR 1	5(1)	SENSOR 3	13(1)	B DESWIT
7.	3(1)	SENSOR 1	5(1)	SENSOR 3	14(1)	B DEALAR
8.	3(1)	SENSOR 1	5(1)	SENSOR 3	12(1)	B DEALAR
9.	4(1)	SENSOR 2	6(1)	SENSOR 4	9(1)	A DESWIT
10.	4(1)	SENSOR 2	6(1)	SENSOR 4	10(1)	A DEALAR
11.	4(1)	SENSOR 2	6(1)	SENSOR 4	8(1)	A DEALAR
12.	5(1)	SENSOR 3	6(1)	SENSOR 4	9(1)	A DESWIT
13.	5(1)	SENSOR 3	6(1)	SENSOR 4	10(1)	A DEALAR
14.	5(1)	SENSOR 3	6(1)	SENSOR 4	8(1)	A DEALAR

----- END OF 3-ORDER FAULTS -----

FIGURE 77 (CONTINUED)



----- START OF 4-ORDER FAULTS -----

TREE VANISHED AT OP 14. NO 4-ORDER FAULTS.

----- END OF 4-ORDER FAULTS -----

HIGHER ORDER FAULT SETS MAY EXIST.

FREQUENCY OF OPERATOR OCCURRENCES IN FAULT SETS

						FAULT SET ORDER			
OP	(NAME)	TYPE	KIND	(NAME)	MODE	TOTAL	1	2	3	4
1	(MAIN BAT)	5	1	(MAIN BAT)	1	1	1	0	0	0
3	(SENSOR 1)	6	3	(SENSOR)	1	8	0	0	8	0
4	(SENSOR 2)	6	3	(SENSOR)	1	8	0	1	7	0
5	(SENSOR 3)	6	3	(SENSOR)	1	8	0	1	7	0
6	(SENSOR 4)	6	3	(SENSOR)	1	8	0	0	8	0
8	(A DEALAR)	5	4	(ALARM BA)	1	5	0	3	2	0
9	(A DESWIT)	6	5	(SWITCH)	1	5	0	3	2	0
10	(A DEALAR)	1	6	(ALARM)	1	5	0	3	2	0
12	(B DEALAR)	5	4	(ALARM BA)	1	5	0	3	2	0
13	(B DESWIT)	6	5	(SWITCH)	1	5	0	3	2	0
14	(B DEALAR)	1	6	(ALARM)	1	5	0	3	2	0

FIGURE 77 (CONTINUED)

means that kind 1, type 5, has 2 values. The first value is treated as a success (critical mode code of 0) and the second value is considered a failure (critical mode code of 1). The reader should refer back to Figure 52, GO2 Output For Alarm System, to verify the intent in this model to treat the first value of 0 generated by this type 5 operator with probability 0.98 as the success mode, and the second value generated, 2, with probability 0.02, as the failure mode.

Figure 77 then records one first-order failure. It is operator #1, the main battery, in a failure mode which will, by itself, cause system failure event 100_2. This information is recorded in the form

1(1) MAIN BAT

where the first "1" is the operator number, and the second "1", in parentheses, is the critical mode code and eight characters of the descriptor are printed, "MAIN BAT".

The printout also shows that there are ten second-order fault sets and 14 third-order sets which cause the system failure event 100_2. The reader may want to convince himself of the validity of these fault sets by referencing Figures 49 and 50 of Chapter 4.

To conclude the FF2 printout a summary table lists each operator which appeared in a fault set with the frequency of occurrence in the various set orders.

Using the identical model, but choosing the selected event to be 100_0, a system premature, the fault sets of Figure 78 were generated. Figure 78 records that there are five second-order combinations of component prematures (critical mode 2) which can cause a system premature. (Again the reader may want to consider the physical configuration to convince himself that the various combinations of sensor prematures will cause a system premature by referencing Figures 49 and 50.

To conclude, the Fault Finder capability is a powerful technique to insure the validity of the GO model, to identify major contributors to system failure, and to consider modifications to enhance system performance. Further information about the Fault Finder can be found in the documents KSC GO Fundamentals and the KSC GO Reference Manual.

```
FF2 DATA FOR ALARM SYSTEM

MODEL: G01 DATA FOR ALARM SYSTEM
KINDS: G02 DATA FOR ALARM SYSTEM

SELECTED EVENTS: SIGNAL NUMBER(VALUE)
   1.   100(  0)

INTER= 1,  FIRST= 0,  LAST= 4,    11 OPS

USER-DEFINED CRITICAL MODES

1 5 2 0 1 $
4 5 2 0 1 $

----- START OF 0-ORDER FAULTS -----

TREE VANISHED AT OP   5.  NO 0-ORDER FAULTS.

----- END OF 0-ORDER FAULTS -----


----- START OF 1-ORDER FAULTS -----

TREE VANISHED AT OP   6.  NO 1-ORDER FAULTS.

----- END OF 1-ORDER FAULTS -----
```

FIGURE 78
FF2 OUTPUT FAULT SETS CAUSING SYSTEM PREMATURE EVENT 100_0

```
----- START OF 2-ORDER FAULTS -----

2-ORDER FAULT SETS
------------------
```

SET	OP(MODE)	NAME	OP(MODE)	NAME
1.	5(2)	SENSOR 3	6(2)	SENSOR 4
2.	4(2)	SENSOR 2	6(2)	SENSOR 4
3.	4(2)	SENSOR 2	5(2)	SENSOR 3
4.	3(2)	SENSOR 1	5(2)	SENSOR 3
5.	3(2)	SENSOR 1	4(2)	SENSOR 2

```
----- END OF 2-ORDER FAULTS -----

----- START OF 3-ORDER FAULTS -----

TREE VANISHED AT OP   6.  NO 3-ORDER FAULTS.

----- END OF 3-ORDER FAULTS -----

----- START OF 4-ORDER FAULTS -----

TREE VANISHED AT OP   6.  NO 4-ORDER FAULTS.

----- END OF 4-ORDER FAULTS -----

NO HIGHER ORDER FAULT SETS EXIST.

FREQUENCY OF OPERATOR OCCURRENCES IN FAULT SETS
```

						FAULT SET ORDER			
OP	(NAME)	TYPE	KIND	(NAME)	MODE	TOTAL	1	2	3	4
3	(SENSOR 1)	6	3	(SENSOR)	2	2	0	2	0	0
4	(SENSOR 2)	6	3	(SENSOR)	2	3	0	3	0	0
5	(SENSOR 3)	6	3	(SENSOR)	2	3	0	3	0	0
6	(SENSOR 4)	6	3	(SENSOR)	2	2	0	2	0	0

FIGURE 78 (CONTINUED)

CHAPTER 6
EFFECT EVALUATION

Introduction

The three Effect Evaluation (EE) programs, EE1, EE2, and EE3, have been created to provide a systematic method of evaluating the effect of data uncertainties upon the probabilities associated with the final events of a GO model. The employment of the EE programs is introduced with a simple example of three components in series.

EE Theory

As a basis for discussion, consider the GO model whose GO chart is shown below in Figure 79.

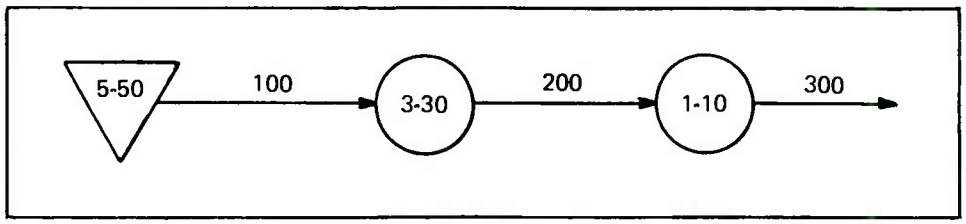


FIGURE 79 GO MODEL FOR EE DISCUSSION

Let infinity (never) = 7.
Let the kind data be as shown in Figure 80.

50	5	3	2	.5	4	.3	7	.2
30	3	.7	.2	.1				
10	1	.75	.25					

FIGURE 80 GO2 DATA FOR EE DISCUSSION

Using GO1, GO2, and GO3 (or a simple hand calculation) we find that signal 300 has the following probability distribution:

VALUE	PROBABILITY
0	.075
4	.1575
2	.2625
7	.505
	Σ = 1.0000

We focus our interest in the event that signal 300 has a value of either 2 or 4. The probability, p, of this event is,

$$p = 0.2625 + 0.1575$$
$$= 0.42$$

Let us now assume that the probabilities used to define kinds 10 and 50 are not fixed but rather are uncertain - that is, are random variables. Our

problem is to estimate the effect of these uncertainties upon p which, of course, now becomes a random variable itself, and which we will designate by P rather than p.

The ultimate solution would be to find the exact probability distribution of P as this would provide all possible information about P - e.g., its mean, its variance, probability ($P \leq 0.9$), etc. This solution would require that the distributions of the random kind probabilities be known, that the formula expressing P as a function of these probabilities be known, and that the analytical machinery for finding the required distribution be available. Unfortunately in most problems of real interest the kind probability distributions are not known, and the formula is almost always of a most formidable nature, at best, and usually requires an exorbitant amount of labor to obtain if it can be obtained at all. In fact, the entire GO concept is based upon the idea of evaluating this function without knowing it explicitly. Consequently, we shall set our sights at something less than the ultimate solution.

What we settle for in the EE procedure are estimates of the mean and variance of P (these will be symbolized by m and v respectively). Using these estimates we can also calculate approximate confidence intervals for the mean of P.

Let us symbolize the variable kind probabilities by P_i (the uppercase letter indicating a random variable), their means by m_i, and their variances by v_i. Also let f be the existing, but generally unknown, system function which relates P to the P_i - e.g., $P = f(P_1, P_2, \ldots)$.

The basic assumption of EE is that f is sufficiently "linear" that reasonable approximations of the mean and variance of P are given by

$$m = f(m_1, m_2 \ldots)$$

and

$$v = \sum_i \left(\frac{\partial f}{\partial m_i}\right)^2 v_i$$

Note that GO3 evaluates the function f for specified values of the m_i without knowing f explicitly.

The partial derivatives in the formula for v are evaluated by subtracting a small positive amount, Δ, from the value of m_i, reevaluating f, and then forming a difference quotient as shown below.

$$\frac{\partial f}{\partial m_i} \approx \frac{f(m_1, \ldots m_i, \ldots) - f(m_1, \ldots, m_i - \Delta, \ldots)}{m_i - (m_i - \Delta)}$$

By choosing Δ small the partial derivatives can be calculated as precisely as desired. The partial derivative of an event with regard to changes in the kind data is known as the sensitivity of the event with respect to kind i.

Once m and v are obtained, several methods could be used to obtain approximate confidence intervals for the mean of P. Because this mean is essentially a probability we use a method due to Easterling. In this method m is treated as an estimate of a binomial proportion which was hypothetically obtained by x "pseudo successes" in n "pseudo trials". If this had indeed been the case we would estimate the proportion mean and its variance by,

$$m = \frac{x}{n} \quad \text{and} \quad v = \frac{m(1-m)}{n}$$

However, we have already obtained estimates for m and v and, consequently, we can now determine n and x by solving these two equations to obtain:

$$n = \frac{m(1-m)}{v} \quad \text{and} \quad x = mn$$

(In practice the values of n and x are rounded up to the next integer.) Using standard procedures the required binomial confidence intervals can now be calculated. (To do so we use an algorithm due to Burstein and Anderson in a series of articles in the Journal of the American Statistical Association; September 1967, 857-61; December 1968, 1413-5; and September 1973, 581-4).

If there are k variable kinds, the function f must be evaluated k+1 times - once to find m and once to find each of the k sensitivities. The entire solution as outlined above can be carried out using just GO1, GO2, GO3, and some hand calculations. In a large problem the hand bookkeeping chores could become quite burdensome and error-prone. The purpose of the EE programs is simply to automate the entire procedure.

There are three EE programs - EE1, EE2, and EE3. EE1, which is executed after a GO1 run, is similar to GO2 and preprocesses the kind data. EE2, uses the operator and kind data provided by GO1 and EE1, and evaluates the system k+1 times (each evaluation using logic identical to that in GO3). EE2 calculates the sensitivity of every elementary final event with respect to each variable kind. Finally, EE3 calculates the required estimates and bounds for any event selected by the user.

When evaluating f for a sensitivity calculation we decrement a kind probability by Δ. Because the several probabilities associated with each kind must sum to unity, this means that one or more of the other probabilities must be incremented. With kind data from two-state operators (type 1 and possibly types 4, 5, 8, and 12) there is no problem. When an operator has more than two states (types 3, 6, 7, and possibly others) the situation is less clear. We have chosen the convention that in such a case only two of the several kind probabilities will be altered - one decremented by Δ and the other incremented by Δ. The one to be decremented will be designated by G and the other one by B. As far as the results are concerned it is immaterial if the G and B are interchanged because if this is done, the signs of the sensitivities will be reversed, but in the computation of v the sensitivities are squared. However to avoid potential problems related to the precision of the computations, it seems safer to assign G to the larger probability (which usually represents some "good" state) and B to the smaller.

The selection of Δ is somewhat arbitrary. In general a smaller Δ gives a better approximation of the partial derivative. However, precision limitations require that the resulting differences be large enough to prevent them from being hidden in the roundoff noise. Our approach is to define a "delta factor" and then calculate as a function of PMIN as follows:

Δ = PMIN x delta factor

The default value of the delta factor is 10, but it may be set to another value by the user.

Let us now return to our example. For the variable kind 10 (type 1) let G_{10} represent the good state probability and B_{10} the dud state probability. Note that $B_{10} = 1.-G_{10}$. Let 0.75 be the mean of G_{10} and let its variance be 0.0001.

For the variable kind 50 (type 5) let G_{50} represent the probability of that the operator signal takes the value 2 and B_{50} the probability of that the signal generated by the operator takes the value 7 (thus the probability of taking the value 4 will remain fixed at 0.3). In this case we see that $B_{50} = 1. - 0.3 - G_{50} = 0.7 - G_{50}$. Let 0.5 be the mean of G_{50} and let its variance be 0.0375.

This model is sufficiently simple that the exact formulas for the final events can be readily found. Let 300_i be the event that the final signal 300 takes on the value i and let $P(300_i)$ be its probability. Then, in terms of G_{10} and G_{50} (which correspond to the m_i in the earlier discussion), we have,

$$P(300_0) = 0.1\ G_{10}$$
$$P(300_2) = 0.7\ G_{10}G_{50}$$
$$P(300_4) = 0.21\ G_{10}$$
$$P(300_7) = 1. - 0.31\ G_{10} - 0.7\ G_{10}G_{50}.$$

These formulas must be evaluated three times. In the first evaluation (the "reference run" which calculates the point estimate of the event probabilities) we replace G_{10} and G_{50} by their means - i.e., $G_{10} = 0.75$ and G_{50} 0.5. In the second evaluation (the "sensitivity" run for kind 10) we use $G_{10} = 0.75 - \Delta$ and $G_{50} = 0.5$. Finally in the third evaluation (the sensitivity run for kind 50) we use $G_{10} = 0.75$ and $G_{50} = 0.5 - \Delta$. Anticipating the actual EE execution in which PMIN=1.E-8, we set Δ = PMINx10 = 1.E-7. The results for the three runs are given below.

EVENT PROBABILITY CALCULATED IN EACH OF THREE RUNS

EVENT	REFERENCE	KIND 10 SENS.	KIND 50 SENS.
300_0	0.075	0.074999990	0.075
300_2	0.2625	0.262499965	0.2624999475
300_4	0.1575	0.157499979	0.1575
300_7	0.505	0.505000066	0.5050000525

From these results the sensitivities themselves are readily found. For example, the sensitivity of the event 300_0 with respect to kind 10 is given by,

$$\frac{0.075 - 0.0749999990}{\Delta} = \frac{0.00000001}{0.0000001} = 0.1$$

The entire set of sensitivities are shown below.

KIND	300 0	300 2	300 4	300 7
10	0.1	0.35	0.21	-0.66
50	0.0	0.525	0.0	-0.525

In this example, because we have the formulas for the event probabilities, the sensitivities could be directly calculated by differentiation. We note that identical results would be obtained because all of the functions are linear with respect to each variable.

As mentioned earlier our interest is in the event that signal 300 takes on a value of either 2 or 4. Denote this "selected event" by E. Then,

$$E = 300_2 \oplus 300_4$$

Where $\oplus$ is the logical (boolean) sum ("OR").

Because the basic events 300_2 and 300_4 are mutually exclusive, the probability of E is given by,

$$\begin{aligned} P(E) &= P(300_2 \oplus 300_4) \\ &= P(300_2) + P(300_4) \end{aligned}$$

Consequently we obtain an estimate of the mean of P(E) by summing the estimates of the means of $P(300_2)$ and $P(300_4)$ which are obtained from the reference run. Hence,

$$m = 0.1575 + 0.2625 = 0.42$$

The variance of P(E) is approximated by

$$v = \left(\frac{\partial P(E)}{\partial G_{10}}\right)^2 v_{10} + \left(\frac{\partial P(E)}{\partial G_{50}}\right)^2 v_{50}$$

where v_i is the variance of G_i (i=10,50). Now,

$$\begin{aligned} \frac{\partial P(E)}{\partial G(10)} &= \frac{\left(\partial P(300_2) + P(300_4)\right)}{\partial G_{10}} \\ &= \frac{\partial P(300_2)}{\partial G_{10}} + \frac{\partial P(300_4)}{\partial G_{10}} \end{aligned}$$

But these terms are simply the sensitivities of the events 300_2 and 300_4 with respect to kind 10. A similar argument can be applied to $P(E)/G_{50}$, and we finally get (recall that $v_{10} = 0.0375$ and $v_{50} = 0.0001$),

$$\begin{aligned} v &= (0.35 + 0.21)^2 \times 0.0375 + (0.0515 + 0.0)^2 \times 0.0001 \\ &= 0.01176 + 0.00002756625 \\ &= 0.0117875625. \end{aligned}$$

Finally, to find a confidence interval for the mean of $P(E)$, we use m and v to calculate the Easterling pseudo values.

$$n = \frac{0.42(1-0.42)}{0.011788} = 20.67 = 21$$

$$x = 0.42 \times 21 = 8.82 = 9$$

From these values Burstein's computer algorithm gives a 95% lower confidence bound of 0.243 and a 95% confidence interval of (0.216, 0.679).

EE Output

This sample problem was executed using the programs GO1, EE1, EE2, and EE3. The output printouts from these programs are shown in Figures 81, 82, 83, and 84. The GO1 printout is recorded in Figure 81. It should be self-explanatory at this point.

The EE1 printout (comparable to GO2 data) is documented in Figure 82. The data for the three kinds, 10, 30, and 50, are listed. Kinds 10 and 50 are variable and the entries G and B communicate this fact to the computer. The kind 30 data is considered to be exact - i.e., there is no uncertainty. It is entered exactly as in GO2.

For kinds which are considered variable there are two ways to enter the data -- either as the values for the mean and standard deviation or as the two parameters (α and β) of a beta distribution. In both cases the data is entered after the normal GO2 kind data (with G and B) replacing certain probabilities considered variable), but before the data terminator ($ or /).

In Figure 82 the data for kind 10 was entered in the form of the two parameters of a beta distribution. In this case $\alpha = 3$ and $\beta = 1$. Consequently, from the expressions for the mean (μ) and variance (σ^2) of a beta distribution,

$$\mu = \frac{\alpha}{\alpha+\beta}; \quad \sigma^2 = \frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)} .$$

the mean is 0.75 and the variance 0.0375.

For kind 50 the data is entered explicitly as the mean and standard deviation, 0.5 and 0.01 respectively.

```
GO1 DATA FOR EE EXAMPLE

INFIN=  7,  SEQ=0,  ERRORS= 25
 BIAS= 750,  OPS=1,  SIGNAL=1

OPERATOR DATA

 OP    DATA
----  ------------------------------------------------------
   1  5 50 100 $ COMP 1
   2  3 30 100 200 $ COMP 2
   3  1 10 200 300 $ COMP 3
///A  0 300 $ FINAL SIGNAL
SIGNAL DATA

         SOURCE OPER.
SIGNAL   NUM TYPE KIND  USING OPERATORS (- IF DELETED AT)
------ ---- ---- ----  ----------------------------------
  100     1    5    50    -2
  200     2    3    30    -3
  300     3    1    10

NUMBER OF OPERATORS     =     3
NUMBER OF SIGNALS       =     3
MAX NUMBER ACTIVE       =     1
INFINITY (NEVER)        =     7

FINAL SIGNALS =   300
```

FIGURE 81 GO1 DATA FOR EE EXAMPLE

```
EE1 DATA FOR EE EXAMPLE


MODEL: GO1 DATA FOR EE EXAMPLE

RECORD KIND DATA
------ ----------------------------------------
    1   10 1 G B 3 1 $ COMP 3
           GOOD=.75000000     ,  BAD=.25000000
    2   30 3 0.7 0.2 0.1 $ COMP 2
    3   50 5 3 2 G 4 0.3 7 B 0.5 0.01 $ COMP 1
           GOOD=.50000000     ,  BAD=.20000000
-------------------------------------------------



USE SUMMARY TABLE. ENTRY=KIND/TYPE(FREQUENCY)
   (FREQUENCY IS NEGATIVE FOR  PERFECT KINDS.)

 10/ 1(     1)    30/ 3(     1)    50/ 5(     1)

KIND SUMMARY
   VARIABLE NUMBER INPUT =  2
   VARIABLE NUMBER USED  =  2
   NONVARIABLE NUMBER INPUT =  1
   NONVARIABLE NUMBER USED  =  1

   3 OP/KIND RECORDS WRITTEN ON  1 FILE RECORDS.
```

FIGURE 82 EE1 OUTPUT DATA FOR EE EXAMPLE VARIABLE KIND DATA

```
EE2 DATA FOR EE EXAMPLE


MODEL: G01 DATA FOR EE EXAMPLE
KINDS:  EE1 DATA FOR EE EXAMPLE

MAXIMUM SIGNAL VALUE (INFINITY) IS     7
MAXIMUM DISTRIBUTION SIZE IS 3000
PMIN=1.0000E-08,  INTER=0
NOVAR=0,  DFACT=  10.0,  KORD=1


-----------------------------------------
FINAL EVENT TABLE (INFINITY =   7)

                SIGNALS AND THEIR VALUES
                ------------------------

EV PROBABILITY      300
-- -------------   ----
 1 7.50000E-02     0
 2 1.57500E-01     4
 3 2.62500E-01     2
 4 5.05000E-01     7

-----------------------------------------

TOTAL PROBABILITY = 1.0000000000
TOTAL ERROR =       0.0000000000
DELTA =  1.0000E-07

-------------------------------------------------------------------------

VARIABLE KIND SENSITIVITIES FOR EVENTS IN ABOVE TABLE
  (CHANGE-IN-EVENT-PROB/CHANGE-IN-THE-KIND-G-PROB)

   EVENT  1         EVENT  2         EVENT  3         EVENT  4
--------------- --------------- --------------- ---------------
 10( 1.000E-01)  10( 2.100E-01)  50( 5.250E-01)  50(-5.250E-01)
 50( 0.000E+00)  50( 0.000E+00)  10( 3.500E-01)  10(-6.600E-01)
```

FIGURE 83 EE2 OUTPUT DATA FOR EE EXAMPLE PARTIAL DERIVATIVES FOR SYSTEM EVENTS

```
EE3 DATA FOR EE EXAMPLE


-------------------------------------------------------------------------------
0.8 0.9 0.95 0.99 $
RUN NUMBER  1  DEFINITION DATA
          0.800     0.900     0.950     0.990

ALL VARIABLE KINDS ARE INCLUDED IN THE ANALYSIS

SELECTED EVENT
-------- -----
300 2 $
300 4 $
EOR
   1.   300(  2)
   2.   300(  4)

                                                          EVENT     CONTRIB TO
KIND   NAME    TYPE USES   G PROB.   G VARIANCE SENSITIVITY EVENT VAR.
---- -------- ---- ---- ---------- ---------- ----------- ----------
  10 COMP 3      1      1 0.75000000 3.7500E-02  5.6000E-01 1.1760E-02
  50 COMP 1      5      1 0.50000000 1.0000E-04  5.2500E-01 2.7563E-05

EVENT PROBABILITY = 4.200000000E-01
VARIANCE ESTIMATE = 1.178756250E-02

EASTERLING INTERVALS
--------------------
     PSEUDO SAMPLE SIZE  =         21
     NUMBER OF SUCCESSES =          9

          CONFIDENCE LOWER BOUND   TWO-SIDED INTERVAL
          ---------- ----------- ----------------------
            0.8000   0.31802642  0.27552971, 0.60029495
            0.9000   0.27552971  0.24268106, 0.64246452
            0.9500   0.24268106  0.21594365, 0.67881441
            0.9900   0.18702532  0.16871519, 0.74872196

-------------------------------------------------------------------------------
```

FIGURE 84

EE2 OUTPUT DATA FOR EE EXAMPLE CONFIDENCE BOUNDS

The EE2 output data is recorded in Figure 83. It first records the possible system events -- just like a typical GO3 run -- then records the partial derivatives of each final output event with regard to each variable kind (10 and 50 in this model). Note that the events are numbered in the final event table, then referenced by number in the sensitivity table. The partial derivative of event 300_2 with regard to kind 50 is 0.525, etc.

The results from program EE3, Figure 84, contains the final output from an Effect Evaluation execution. The user has the capability to select various confidence levels and to select the system event whose probability of occurrence is to be bounded.

In Figure 84, one- and two-sided bounds for the compound event 300_2 or 300_4 at confidence levels of 0.8, 0.9, 0.95, and 0.99, are requested. The mean, variance, partial derivative, and contribution to the variance for both kind 10 and kind 50 are recorded in table form. The mean probability of occurrence of the event 300_2 or 300_4 is 0.42 and the variance estimate is 0.0118. From this data the pseudo sample size and number of successes of 21 and 9 are calculated. These values are then used with binomial theory to calculate the confidence bounds at the specified levels.

For example a 95% one-sided lower bound on the event mean, estimated at 0.42, is 0.243. The 95% two-sided confidence interval is 0.216 to 0.679 as the table shows. Note that as the confidence increases the intervals increase in length and the lower bound decreases. One- and two-sided confidence bounds for each selected confidence level is provided in the final table.

We have explained and demonstrated by means of an example the theory and application of the Effect Evaluation codes to treat data uncertainties and place bounds upon the probabilities of occurrence of system events. The method is easy to use, extremely flexible, and is applicable to any system regardless of its complexity. Further details about the EE concept and programs are available in the KSC GO Fundamentals and GO Reference Manuals.

CHAPTER 7
CONCLUSIONS AND RECOMMENDATIONS

Conclusions

GO is a computerized methodology for performing RAM assessments. It has widespread application for performing safety, reliability, availability, maintainability, and risk assessments. It is available to all users.

The GO methodology provides a disciplined approach to perform comprehensive system assessments. The procedure has been qualified, validated and accepted as a powerful technique which provides correct results. The GO software incorporates all the necessary classical algorithms to accurately and comprehensively calculate the probabilities of occurrence of system events as a function of the probabilities and sequence of occurrence of component events.

Use of the GO methodology facilitates computerized reporting and eliminates much time-consuming, tedious, error-prone hand processing required by other assessment methods. GO models are compact and easy to follow because they closely track original schematics or flow diagrams. The models are easily developed by engineers without extensive specialization and the computer software is easy to use with minimal training. Consequently, the GO methodology eliminates constraints with which other methods are encumbered.

The efficiency of the GO procedure results from the use of standardized logical operators (types), supertypes to model replicated equipment configurations, and the optimized software. This efficiency, both in analyst time for model development and in computer manipulation time, permits extensive and exhaustive system assessments with reduced costs and scientific labor.

Use of the GO methodology does not require additional sampling or testing programs. To employ it, no new data requirements are imposed, but existing data is used consistently and effectively to comprehensively assess system performance.

The GO code is generally applicable. GO models can be developed for almost any system.

The GO procedure provides for automatic fault identification and treatment of data uncertainties without additional modeling tasks. Sensitivities and confidence bounds on the probabilities of occurrence of system events can be generated automatically.

Recommendations

The GO software is currently available at the ARRADCOM, Dover, New Jersey, installation. It is recommended that other commands performing RAM assessments obtain the GO software and establish a training program to effect its immediate use and application.

DISTRIBUTION LIST

Commander
U.S. Army Armament Research and
Development Command
ATTN: DRDAR-LCU
DRDAR-LCN
DRDAR-QAN (5)
DRDAR-LCN-DP
DRDAR-QA
DRDAR-QAP
DRDAR-QAT
DRDAR-QAR
DRDAR-QAF
DRDAR-QAC
DRDAR-QAS
DRDAR-TSS (5)
DRDAR-SF
Dover, NJ 07801

Commander
U.S. Army Armament Materiel
Readiness Command
ATTN: DRSAR-MAY-A
DRSAR-MAY-B
Dover, NJ 07801

HQ, Department of the Army
ATTN: DALO-SMD
DAMO-ODC
DARD-DDM
DAFD-SDX
DAPE-SD
DAMO-NCS
DAMA-CSS-N
Washington, DC 20310

Commander
Department of the Army
ATTN: FDSB-ND
Bldg 2073 North Area
Ft. Belvoir, VA 22060

U.S. Department of Energy
Director of Military Applications
ATTN: SE&EA
Washington, DC 20545

U.S. Department of Energy
Albuquerque Operations Office
ATTN: QA Division
P.O. Box 5400
Albuquerque, NM 87115

Department of Energy
Division of Military Applications
Germantown, MD 20545

Commander
U.S. Army Materiel Development and
Readiness Command
ATTN: DRCQA
DRCNC
DRCDE-DM
DRCPM-NUC-W
DRCSF-E
5001 Eisenhower Avenue
Alexandria, VA 22333

Commander
U.S. Army Materiel Development and
Readiness Command
ATTN: DRCPM-NUC-M
Rock Island, IL 61299

Commander
U.S. Army Materiel Development and
Readiness Command
ATTN: DRCPM-NUC-HL
Huntsville, AL 23809

Commander
U.S. Army Materiel Development and
Readiness Command Field Office
ATTN: DRXFO
Kirkland Air Force Base
Albuquerque, NM 87115

Commander
U.S. Army Armament Materiel
Readiness Command
ATTN: DRSAR-QA
DRSAR-ASN
DRSAR-MA
DRSAR-PP
DRSAR-LEP-L
Rock Island, IL 61299

Director
Ballistics Research Laboratory
U.S. Army Armament Research and
Development Command
ATTN: DRDAR-TSB-S
DRXBR-XSG
DRXBR-EB-FT
Aberdeen Proving Ground, MD 21005

Commander
U.S. Army Combat Development Command
ATTN: CDCCD-F
Ft. Belvoir, VA 22060

Commander
U.S. Army Missile Command
ATTN: DRSMI-SSB
DRSMI-Q
DRCPM-LC
DRSMI-U
Redstone Arsenal, AL 35809

Department of the Army
Ballistic Missile Defense Systems Command
ATTN: BMDSC-TTP
P. O. Box 1500
Huntsville, AL 35807

Commander
U.S. Army Test and Evaluation Command
ATTN: DRSTE-FA
Aberdeen Proving Ground, MD 21005

Commander
U.S. Army Training and
Doctrine Command
ATTN: ATCD-CF-SS
Fort Monroe, VA 23651

Commander
Naval Sea Systems Command
Naval Ammunition Production
Engineering Center
Naval Weapons Support Center
Code SEA-642521
Crane, IN 47522

Commander
U.S. Army Air Defense Command
ATTN: ADGPS-M
Ent Air Force Base
Colorado Springs, CO 80912

Commander
U.S. Army Forces Command
ATTN: AFOP-TS
Fort McPherson, GA 30330

Director
Weapons System Evaluation Group
400 Army-Navy Drive
Arlington, VA 22203

Director
Defense Nuclear Agency
ATTN: OPSD
OPSM
STNA
Washington, DC 20305

Commander
U.S. Army Nuclear Agency
ATTN: MONA-SU
Fort Belvoir, VA 22060

Commander
U.S. Army Nuclear Agency
ATTN: MONA-MS
Fort Bliss, TX 79916

Director
Defense Nuclear Agency
ATTN: OPTO
Hybla Valley Federal Building
6801 Telegraph Road
Alexandria, VA 20305

Commander
U.S. Army Operational Test and
Evaluation Agency
ATTN: CSTE-STO-A
Falls Church, VA 22041

Commander
U.S. Army Nuclear and Chemical Agency
ATTN: MONA-MS
7500 Backlick Road
Building 2073
Springfield, VA 22150

Department of the Army
Assistant Chief of Staff for
Force Development
ATTN: FOR-CM-NU
Washington, DC 20310

Director
Defense Research and Engineering
Department of Defense
Washington, DC 20301

Project Manager
Nuclear Munitions, Field Office
ATTN: DRCPM-NUC-AFO
P. O. Box 5390
Kirkland Air Force Base
Albuquerque, NM 87185

Commander
Field Command, Defense Nuclear Agency
ATTN: FCPSQ
Kirkland Air Force Base
Albuquerque, NM 87185

Sandia National Laboratories
ATTN: 7220
Albuquerque, NM 87185

Los Alamos National Scientific
Laboratory (LANSL)
P. O. Box 1663
Los Alamos, NM 87545

Lawrence Livermore National
Laboratory
University of California
P. O. Box 808
Livermore, CA 94550

Vice President
Sandia National Laboratories
P. O. Box 969
ATTN: D. Bohrer
W. Gordon
Livermore, CA 94550

Commander
Harry Diamond Laboratories
ATTN: DELHD-ED
2800 Powder Mill Road
Adelphi, MD 20783

Commander
U.S. Army Materials and Mechanics
Research Center
ATTN: DRXMR-KB
Watertown, MA 02172

Commander
U.S. Army Aviation School
ATTN: ATZQ-D-MR
Ft. Rucker, AL 36362

Commander
U.S. Army Chemical School
ATTN: ATZN-CM-CDM
Ft. McClellan, AL 36201

Commander
U.S. Army Watervliet Arsenal
ATTN: SWEWV-RDD-SP
Watervliet, NY 12189

Director
U.S. Army Materiel Systems
Analysis Agency
ATTN: DRXSY-RE
DRXSY-MP
Aberdeen Proving Ground, MD 21005

Commander
Naval Sea Systems Command
ATTN: SEA-9931G

Commander
U.S. Army Engineer School
ATTN: ATZA-CDI
Ft. Belvoir, VA 22060

Commander
U.S. Army Missile and
Munitions School
ATTN: ATSK-CD-CS
Redstone Arsenal
Huntsville, AL 35809

Commander
U.S. Army Ordnance School
ATTN: ATSL-CD-MS
Aberdeen Proving Ground, MD 21005

Commander
U.S. Army Quartermaster School
ATTN: ATSM-CM-M
Ft. Lee, VA 23801

Commander
U.S. Army Signal School
ATTN: ATZH-CD-IL
Ft. Gordon, GA 30905

Commander
U.S. Army Transportation School
ATTN: ATSP-CD-MSR
Ft. Eustis, VA 23604

Commander
U.S. Army Field Artillery School
ATTN: ATSF-CD-W
Ft. Sill, OK 73503

Director
DARCOM Intern Training Center
ATTN: DRXMC-ITC-E
Red River Army Depot
Texarkana, TX 75507

U.S. Army Logistics Management
Center (ALMC)
Fort Lee, VA 23801

Johns Hopkins University
Applied Physics Laboratory
8621 Georgia Avenue
Silver Spring, MD 20910

APL
Johns Hopkins University
Applied Physics Laboratory
Pershing Program
Johns Hopkins Road
Laurel, MD 20810

National Aeronautics and Space
Administration
Goddard Space Flight Center
Greenbelt, MD 20771

Kaman Sciences Corporation (10)
1500 Garden of the Gods Road
Colorado Springs, CO 80933

Mr. Nicholas Zuck
140 Canoebrook Parkway
Summit, NJ 07901